DOCUMENT RESUME

ED 028 644 EM 007 141

By-Frye, C. H.; And Others

Interim User's Guide to PLANIT: The Author-Language of the Instructor's Computer Utility. Technical Memorandum.

System Development Corp., Santa Monica, Calif.

Spons Agency-National Science Foundation, Washington, D.C.

Report No-NSF-C557; TM-3055-000-03

Pub Date 16 Oct 68

Note-94p.; This manual is the revised edition of "User's Guide to PLANIT" by S. L. Feingold and C. H. Frye, October, 1966; SDC Document TM-3055/000/01

EDRS Price MF-$0.50 HC-$4.80

Descriptors-*Computer Assisted Instruction, *Computer Oriented Programs, *Computer Programs, Constructed Response, *Curriculum Development, Feedback, *Manuals, Optional Branching, Programed Instruction, Programing, Time Sharing

Identifiers-PLANIT, *Programming Language for Interactive Teaching

PLANIT (Programming Language for Interactive Teaching) is a general purpose teaching system that allows a lesson designer to enter course content into the computer for use as a teaching device. The user (lesson designer or student) communicates with the system via a keyboard. Interacting with PLANIT, he can build and edit lessons, present lessons, and perform computations. PLANIT employs a four mode system: Command Mode, Lesson Building Mode, CaSc Mode, and Executive Mode. Through phonic comparison, keyword matching, and algebraic matching, it also provides service functions for evaluating student answers that depart from the expected response. This manual is the revised edition of the user's guide to the version of PLANIT that operates on the System Development Corporation Q-32 Time-Sharing system. (Author/BB)

ED028644

U.S. DEPARTMENT OF HEALTH, EDUCATION & WELFARE
OFFICE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL OFFICE OF EDUCATION POSITION OR POLICY.

TM-3055/000/03

TECHNICAL MEMORANDUM

(TM Series)

The work reported herein is in partial fulfillment of the National Science Foundation Contract #C557.

Interim User's Guide to PLANIT:

The Author-Language of the Instructor's Computer Utility

By
C. H. Frye, F. D. Bennik,
and S. L. Feingold

16 October 1968

SYSTEM
DEVELOPMENT
CORPORATION
2500 COLORADO AVE.
SANTA MONICA
CALIFORNIA
90406

SDC

EM007141

A-2348 4/65

ABSTRACT

PLANIT (Programming LANguage for Interactive Teaching) is a general-purpose teaching system that allows a lesson designer to enter course content into the computer for use as a teaching device.

The user (lesson designer or student) communicates with the system via a keyboard device linked by either TWX, Telex, or telephone to the computer. Interacting with PLANIT, the user can build and edit lessons, present lessons, and perform computations.

FOREWORD

This manual is the preliminary language specification for the author-language, PLANIT, as it will be implemented in the Instructor's Computer Utility. It is a revised edition of the user's guide for the version of PLANIT that operates on the System Development Corporation Q-32 Time-Sharing system.*

---

*Feingold, S. L. and Frye, C. H. User's Guide to PLANIT, SDC Document TM-3055/000/01. October, 1966.

TABLE OF CONTENTS

Page

INTRODUCTION 7
GENERAL INFORMATION 7
(#CO) THE LESSON BUILDING MODES 12
    (Q) THE QUESTION FRAME 13
    SPECIFICATIONS FOR FORMULAS, KEYWORD AND PHONETIC 26
    (Q) THE DICHOTOMOUS FRAME (POS/NEG) 28
    (M) THE MULTIPLE CHOICE FRAME 29
    (D) THE DECISION FRAME 30
    (P) THE PROBLEM FRAME 36
    (C) THE COPY FRAME 43
    SPECIFICATIONS FOR LINKING LESSONS 44
(#P, I, D) FRAME EDITING 46
(#GET) GETTING A LESSON OR STUDENT'S RECORDS 49
(#SAVE) SAVING A LESSON 50
(#EX) EXECUTING A LESSON 51
(#RESTART) ERASE AND RESTART LESSON BUILDING 51
(#DISP) DISPLAY STUDENT'S RECORDS 52
(#CALC, ←) THE CALC MODE 53
CONDENSED LESSON BUILDING INFORMATION: A SUMMARY 81
    LEGAL COMMANDS (#) 81
    THE FRAME TYPES P/Q/M/D/C 83
APPENDIX A: USING CALC STATEMENTS IN THE FRAME 90
APPENDIX B: AUTOMATIC TEXT REFORMATTING 93
APPENDIX C: TECHNIQUE USED FOR EVALUATING ALGEBRAIC EXPRESSIONS . . . 94
APPENDIX D: PHONETIC ENCODING 95
APPENDIX E: ADDITIONAL CAPABILITIES OF THE PROBLEM FRAME 96

INTRODUCTION

PLANIT is a system employing a flexible language designed for computer human interaction. It will allow a lesson designer (LD) to enter questions, and specify answers and actions to take as a function of the student's answers. The language includes statistical functions and a mathematical capability that will allow an instructor to present problems, generate sample data for those problems, and query and evaluate the student's response in terms of the samples generated and the statistical routines that operate on them.

Provision is made for decision branching, recording, and course editing.

The CALC mode of PLANIT by itself can be used as a highly sophisticated calculator for defining and evaluating mathematical functions.

PLANIT also provides service functions for evaluating student answers that depart from the expected response by making PHONETIC comparison, KEYWORD match, equivalent ALGEBRAIC matching, etc.

GENERAL INFORMATION

There are four modes of operation:

COMMAND MODE

This mode can only be used by the LD. It is through this mode that all lesson editing is done. The LD can print frames, delete frames, and insert frames. He can also display student's records, save or get lessons. This is also the mode in which PLANIT starts off. By using the appropriate command the LD can enter any of the other three modes, i.e., LESSON BUILDING mode, EXECUTION mode, or the CALC mode.

LESSON BUILDING MODE

This mode is used to build lesson segments.* Lessons (segments) are composed of frames; frames are composed of groups; groups are composed of lines of information such as textual material, questions, anticipated answers, actions, etc.

CALC MODE

This mode is designed for use by both the LD and the student and provides a powerful computing capability. Explicit arithmetic expressions can be evaluated (i.e., the indicated arithmetic can be performed). Mathematical functions may be defined and evaluated. There is also a variety of "stored" functions and primitives for use in arithmetic expressions or in function definitions: e.g., FACT(N) stands for the factorial of N, ZTOP(X) stands for normal probability integral with upper limit X and lower limit at minus infinity, etc. Matrices may be defined also and the elements of these matrices can be generated through function definition and execution. While taking a course, the student can use this mode in solving the problems presented by the lesson. The lesson designer may use this mode to define functions to be made available to the student.

---

*A lesson segment is the smallest unit of data that can be saved or retrieved from disk or tape. A lesson is a grouping term and is composed of one or more lesson segments.

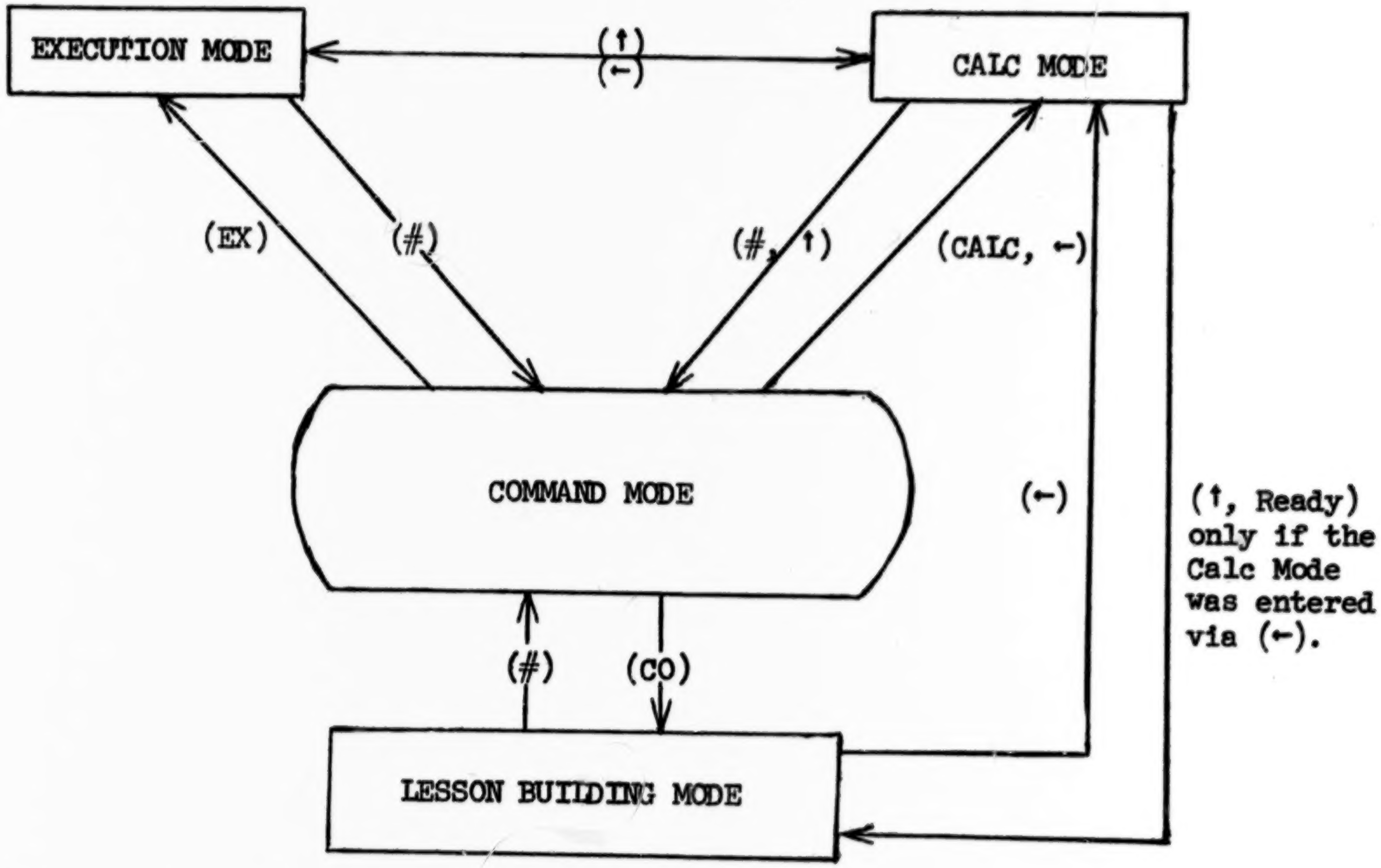
EXECUTION MODE

This is the mode used for presenting lessons to students. Once a lesson is built, it (or parts of it) can be executed. If PLANIT is presenting lessons where the answers require computation, the user may interrupt the EXECUTION mode to enter the CALC mode in order to compute his answers. He then returns to the lesson EXECUTION mode to insert his answers for PLANIT to judge. Alternation between these two modes is effected by the user taking a single button action.

Figure 1 shows the four modes and the special symbols used to alternate among them.

In order to use PLANIT to its full capacity, it is important to understand its four modes of operation, namely, the LESSON BUILDING mode, the EXECUTION mode, the CALC mode, and the COMMAND mode. The LESSON BUILDING mode is the mode within which all lessons are constructed. Lesson construction consists of instructing PLANIT about any text that is to be presented to the student and about the actions to be taken by PLANIT as a function of the student's response to the presentation. Lesson execution consists of the actual presentation of text and implementation of specified actions. Having built a lesson, the LD may want to make changes to it. He may want to print parts of the lesson, insert into parts of the lesson, or delete parts of the lesson. He can do this by entering the COMMAND mode and utilizing the appropriate commands for printing, inserting, or deleting (see the EDIT Commands, page46).

At any time, the LD can enter the CALC mode by typing in the letters CALC prefaced by the hash mark (# - upper case 3). Once in the CALC mode, the LD can perform any CALC statements allowable there. Having done this, he may desire to continue building his lessons and can do so by typing in the hash mark followed by the letters CO. This will get him back into the LESSON BUILDING mode, and he can continue with the next frame from where he left off. The LD can also enter the CALC mode directly from the LESSON BUILDING mode by the left arrow (←). This puts him in the CALC mode as the LD. The advantage is that the lesson is exactly as the LD left it when he returns via the up arrow (↑). When the student is interacting with the lesson he can go into the CALC mode by striking the left arrow and execute any CALC statement allowable to him. The allowable CALC statements to the student


EXECUTION MODE
CALC MODE
COMMAND MODE
LESSON BUILDING MODE
(↑)
(←)
(EX)
(#)
(#, ↑)
(CALC, ←)
(←)
(↑, Ready) only if the Calc Mode was entered via (←).
(#)
(CO)

#	The hash mark is used to enter the COMMAND mode (PLANIT prints UNRECOGNIZABLE CHARACTER if the student attempts to use it).
CO	The CO command is used to enter the Lesson Building mode to COmmence or COntinue building lessons.
EX	The EX command is used to enter the EXecution mode.
↑	The up arrow is used to return to the mode from which the CALC mode was entered.
←	The left arrow is used by the student, or the ID, to enter the CALC mode.
CALC	The CALC command is used by the ID to enter the CALC mode.

Figure 1 Mode Selection Commands

are a subset of all the CALC statements allowable to the LD. The LD always has available to him all CALC statements and all items (see CALC 1.0) used by the student as well. Clever use of this technique can improve the LD's control over what the student is doing. The student in the CALC mode, having made some CALC statements, now desires to return to the lesson proper and enter his answer. He can do this in one of two ways. He can type in READY. PLANIT will respond with ENTER YOUR ANSWER. At this time he can type in the answer that he feels is correct. Another way of leaving the CALC mode is by typing the up-arrow. Using the up-arrow allows the student to follow it immediately with his answer. PLANIT will then evaluate his answer and go on to subsequent frames. In short, although four distinct modes of operation are available, they are integrated into a single system called PLANIT. Actually the LD is the only one who can alternate among four modes at his discretion. The student can only go back and forth between the EXECUTION and the CALC mode.

At any time, if the user (LD or student) does not know where he is, he can always enter the question mark (?) and PLANIT will respond by telling him what is expected of him at that time.

Remember:

1. # is used to get into the COMMAND mode. This hash mark can be followed immediately by any command, e.g., #CALC, #CO, #DISP, #SAVE, etc.

2. ← (left arrow) is used by the student or LD to get into the CALC mode.

3. ↑ (up arrow) is used (LD or student) to return to the previous mode from the CALC mode. This symbol can be followed immediately by a command for the previous mode.

4. READY returns to the previous mode from the CALC mode.

5. CALC puts you into the CALC mode (you must be in the COMMAND mode to use this command).

Once in any mode, the symbols are not needed to exercise any command legal in that mode; rather, the symbols are used to change modes.

We have typed in GO and PLANIT has responded with the asterisk (*) indicating to the user that it is ready to accept his first edit command.

PLANIT interacts with the LD on the TTY. In this manner the LD always knows what is expected of him at any time during the construction of a lesson.

PLANIT will supply information about itself to help the LD build his lesson. This help is provided at one of two levels: verbose or concise. PLANIT always starts at the concise level. The LD can request verbose interaction at this time by typing in ←BE VERBOSE. PLANIT will respond with WILL DO. The LD would now have to return to the COMMAND mode by typing in READY. (Remember that the ← puts the user in the CALC mode; it is in that mode that commands such as BE VERBOSE or BE CONCISE are accepted.)

Still, we may be faced with the problem of too much information or too little information, depending upon the level (verbose or concise). As a compromise, we employ a technique: the question mark (?) which is used at the concise level.

By utilizing the ?, we can always get an elaboration of the last message typed out by PLANIT. Thus, one normally works at the concise level (unless he is completely new to PLANIT), knowing he can always get an elaboration of the last message typed by just entering the ?.

We will now examine in detail the various commands available to the LD when he uses PLANIT:

- to build a lesson,
- to make changes in the lesson,
- to present the lesson to himself (in the role of student),
- to perform mathematical calculations,
- to GET and SAVE the lesson.

We therefore select the appropriate command from the legal commands, which are as follows:

	#CO	to COntinue building the lesson
Edit Commands	P	to Print all or parts of the lesson
	I	to Insert or change parts of the lesson
	D	to Delete parts of the lesson
	#GET	to GET or retrieve a previously built lesson
	#SAVE	to SAVE a lesson
	#RESTART	to clear out all data tables and start over
	#EX	to EXecute a lesson
	#CALC	to use the CALC mode of PLANIT as a lesson designer
	←	to use the CALC mode of PLANIT as a student or lesson design
	#DISP	to DISPlay the student's records

(#CO) THE LESSON BUILDING MODE

When LD receives *, requesting his first command, he may type CO which will transfer him to the Lesson Building mode. He may then specify the type of frame he wants to build.

Generally speaking, all lesson material is entered into the program as a "Frame." Frames consist of units of information called groups. The unit of information within the group is a line.

Throughout this discussion (and in practice), all data to be entered by the LD follows an asterisk typed out by PLANIT. All capitalized letters or numerical data not prefixed by the asterisk are information typed out by PLANIT. The simplest way to explain the language is by giving examples of each frame type. Numbers appearing to the left of any line are actually printed out by the computer denoting a group type within the frame. These numbers will also be used as references in the discussions that follow each frame.

The LD types CO and PLANIT responds with P/Q/M/D/C, and prints an asterisk (*) to signify that an input is expected.

We are at the concise level and so we get an abbreviation of what PLANIT is asking us, namely, to select the type of frame we wish to build. Even though we are at the concise level, we can still get an elaboration of PLANIT's last message by typing in the question mark (?). Having done so, PLANIT responds with (P)ROBLEM/(Q)UESTION/(M)ULTIPLE CHOICE/(D)ECISION/(C)OPY.

There are five basic types of frames. Although the user can select and use any frame type immediately, the frame types are listed here in the suggested order for a new user to introduce himself to the language. The frame types are:

(Q) Question frame

(M) Multiple choice frame

(D) Decision frame

(P) Problem frame

(C) Copy frame

As his next input the LD merely types in the first letter of that frame type, for example, Q.

(Q) THE QUESTION FRAME

The question frame is extremely flexible and would probably be used mostly for general interaction with the student.

This frame is used to present textual information. It can also be used to present and evaluate constructed response and dichotomous answer questions.

A new user should look at the constructed response type first. This version of the Q-frame may be the only one a LD needs for his lessons.

Constructed Response

(Remember, all information not prefaced by an asterisk is typed out by PLANIT. All information prefaced by the asterisk is typed in by the LD.)

Simple use of the Q-frame. The LD types in Q, PLANIT responds with:

```
FRAME 1.00 LABEL=*HIST

2.  TEXT.
*?
2.  SPECIFY QUESTION.
*WHO INVENTED THE ELECTRIC LIGHT?
*

3.  ANSWERS.
*A+THOMAS EDISON
*B ALEXANDER BELL
*

4.  ACTIONS.
*A F:THATS VERY GOOD B:3
*B R:HE INVENTED THE TELEPHONE, TRY AGAIN...
*-R:
*
```

Explanation of Frame 1

All frames are automatically numbered. Here the LD chooses to label the frame HIST. (If he chose not to label the frame, he would merely strike the space bar followed by the carriage return which advances him to Group 2.)

2. TEXT. The LD was not sure what TEXT stood for and typed in the question mark (?). PLANIT immediately elaborates with 2. SPECIFY QUESTION. The LD then types in his question "Who invented the electric light?" PLANIT returns with an asterisk waiting for more lines of input. PLANIT has no way of knowing

when the LD is through inputting the text or question and so returns with an asterisk for each next line. The LD ends the group by striking the space bar once followed by the carriage return key.

3. ANSWERS. Having entered the space and carriage return, PLANIT returns with "3. ANSWERS." A stroke of the ? would have produced the elaboration "3. SPECIFY ANSWER." The LD now enters all the anticipated answers he will accept. In doing so, he tags the first one A and the second one B. Note the + next to the tag A which indicates to PLANIT that this is the correct answer. Having finished entering answers, the LD indicates this by a space and carriage return (CR) and we go on to the action group.

4. ACTIONS. Stands for SPECIFY ACTIONS which the LD could have learned about with a stroke of the ? key. In this group we specify what we want to do next depending upon which answer the student gives.

Action Commands

There are four types of commands in the action group: F:, R:, C:, B:.

F: means what follows is the feedback message that is to be presented to the student. If no message follows F:, then PLANIT will choose one from its feedback table. PLANIT has a list of correct and incorrect type feedback messages, and will select one from the appropriate list (randomly according to whether or not the answer associated with this action was correct or incorrect). This allows the LD to enter F: by itself knowing the student will not always get "YES, YES or NO, NO, etc." every time he enters an answer. The response will normally be one word messages, e.g., YES, FINE, CORRECT, CHECK, or NOT TRUE, NO, WRONG, etc.

R: operates the same as F:, except that R: also instructs PLANIT to wait for another answer (i.e., do not print out the question, just wait for another answer). Finally, R: by itself means print out the fixed message, "WRONG, TRY AGAIN" (then wait for another answer).

C: by itself means print out the fixed message "THE CORRECT ANSWER IS" followed by the correct answer as indicated by the first plus (+) in Group 3. Characters following a C: cause a different interpretation, namely that of a CALC statement. For example C: CONT=CONT+1. This causes PLANIT to evaluate "CONT=CONT+1" in the CALC mode. Here the LD would be incrementing CONT by one (see CALC 2.3.1 and 2.4).

B: means branch (e.g., B:3 means branch to Frame 3). All frames are numbered. They can also be labeled, and a branch can be made to any numbered or labeled frame. In addition, "B:LSNAM" where LSNAM is the name of another lesson, means that the lesson LSNAM will now be brought into PLANIT and executed. The student will never know that another lesson is being called except for some delay in execution of the next frame. This delay will be apparent to the student only when the called lesson must be accessed from tape. Upon completion of the lesson LSNAM, PLANIT will continue with the next frame in the original lesson. Similarly "B:PGM" where PGM might be the name of a compiled binary program used as a subroutine. Again the calling lesson will continue from where it left off when control is returned. One can also specify tape reel number of a lesson, e.g., B:LSNAM REEL (1234). PLANIT would request operator intervention to mount the designated reel if the lesson was not on disc.

B: by itself means return to the calling lesson (i.e., if lesson AA called on lesson BC "B:BC", then PLANIT would return from lesson BC when PLANIT comes across "B:"); i.e., B: with no label or number following it. If CNT is some item and contains a number (put there in CALC perhaps), then B:CNT means branch to the frame number contained in CNT: e.g., if LINK contained the value 20, then B:LINK means branch to Frame 20 and execute as per the instructions in Frame 20. Frames mentioned in a branch need not exist unless that branch is executed during the running of the lesson. If it is executed and no frame exists, then PLANIT will interrupt the lesson and print out an error message stating the illegal branch, the frame, group, and line number where it was requested.

In the sample frame above, the first line in Group 4 is read as: If the student gave Answer A (i.e., THOMAS EDISON), then, print out the message: THATS VERY GOOD and then branch on to Frame 3.

The second line is interpreted as: If the student gave Answer B, then print out: HE INVENTED THE TELEPHONE, TRY AGAIN... then wait for another answer.

The third line indicates an action to be performed if the student's answer did not correspond with either A or B; i.e., for any unanticipated answer (actions prefaced by the minus sign), repeat the frame. (In this case note that nothing appears after the R:. This means print out the fixed message: WRONG, TRY AGAIN and wait for another answer.)

The LD terminated the frame by striking the space bar and the carriage return key (CR).

Remember, the space bar and CR alone on any line terminates the group. The dollar sign "$" and CR alone on any line terminates the frame. If, for example, the LD (in the middle of Group 3) decided to end the frame and go onto the next frame, he would only have to enter the $ and CR alone on a line and PLANIT would respond with:

```
P/Q/M/D/C.
*Q

FRAME 2.00 LABEL=*MATH
```

```
2.  TEXT.
*LET'S SEE WHAT YOU REMEMBER ABOUT TEMPERATURE.  USING F FOR DEGREES
*FAHRENHEIT AND C FOR DEGREES CENTIGRADE, WRITE THE FORMULA FOR
*CONVERTING FROM DEGREES FAHRENHEIT TO DEGREES CENTIGRADE.\
*HINT:  F=9*C/5+32  CONVERTS FROM CENTIGRADE TO FAHRENHEIT.
*

3.  ANSWERS.
*0 FORMULAS ON
*A+C=(5/9)*(F-32)
*B F=9*C/5+32
*C C=(5/9)*F-32
*

4.  ACTIONS.
*A F:   B:7
*B R:YOUR ANSWER IS THE SAME AS THE ONE I GAVE YOU, TRY AGAIN...
*A F: NOW YOU'VE GOT IT.  B:15
*B R:YOU'RE STILL CONVERTING FROM CENTIGRADE TO FAHRENHEIT, TRY AGAIN...
*BC C:  F:NOTE THE DIFFERENCE.  B:OUT
*-R:
*-C:
*
```

Explanation of Frame 2:

The LD labels this frame MATH.
2. TEXT. The LD enters his question. Notice the back slash "\" after CENTIGRADE on the 3rd line. This instructs PLANIT to skip a line after printing out centigrade. To a student taking the lesson, the question would appear exactly as typed (remember, the leading asterisks were not typed) with the exception of the "\" which causes an extra carriage return.

The "\" can be used anywhere in Group 2. Two in a row "\\" means skip two lines, etc.

3. ANSWERS. This group illustrates the algebraic matching ability of PLANIT (turned on by the expression: 0 FORMULAS ON). The student can type in any equivalent algebraic form of the correct answer and get full credit for it; e.g., C=(F-32)*(5/9), C=5*(F-32)/9, C=(5*F-160)/9, etc., are all equivalent and therefore acceptable forms. If FORMULAS is not turned on, or is turned off by the expression: 0 FORMULAS OFF, then only the exact form as typed in by the LD would be looked for in the matching.
We do not wish to mislead the user; this is not true symbol manipulation. We merely employ a technique which (in part) includes performing algebra on the student's answer. For details on the technique see AppendixC .

Note: (5/9)*(F-32)=C is not considered an equivalent form to PLANIT. Naturally the matching technique is not restricted to correct answer alone but works for all anticipated answers in Group 3.

4. ACTIONS. This group illustrates repeated use of a frame.
The FIRST time through this frame, if the student's answer corresponded to:

A--he would receive a (randomly selected) affirmative feedback message followed by a branch to Frame 7.

B--he would receive the feedback message "YOUR ANSWER IS THE SAME AS THE ONE I GAVE YOU, TRY AGAIN...".
(Remember the R: means wait for another answer.)

C--he would receive: NOTE THE DIFFERENCE. THE CORRECT ANSWER IS: C=(5/9)*(F-32) followed by a branch to the frame whose label is OUT.

If his answer did not correspond to either A, B, or C (unanticipated response) he would have received: WRONG, TRY AGAIN.

The SECOND time through the frame, if the student's answer corresponded to:

A--he receives: NOW YOU'VE GOT IT., followed by a branch to Frame 15. This action is performed regardless whether or not his first answer corresponded to A. In other words, the nth repetition of a lettered (or numbered) action, counting from the top of Group 4, is performed the nth time through the frame (provided, of course, that his answer corresponds to that lettered or numbered action). Again the action chosen is dependent only upon the student's answer and the number of times he has been through the frame and not upon what answer he gave before in this frame. (However, for an alternate means of sequencing action execution, see the RELATED command on the following pages.)

B--he receives: YOU'RE STILL CONVERTING FROM CENTIGRADE TO FAHRENHEIT. TRY AGAIN...

C--he receives the same as he would have gotten the first time through the frame had his answer corresponded to C then.

If no match (2nd unanticipated answer) he receives:
THE CORRECT ANSWER IS: C=(5/9)*(F-32). PLANIT would then go on to the next frame in the sequence.

The THIRD (or more) time through the frame, if the student's answer corresponded to:

A, same for A the second time through.

B, same for C any time through the frame.

C, same as before.

Unanticipated answer, same as second time through.

Note: F: C: R: B: appearing on one line are done in the order shown. If the same command appears more than once on the same line, the priority is from left to right. If "B:" appears more than once on any line, the left-most branch will always be executed.

If there is no Group 3, then there will be no pause for the student's answer.

Group 4 commands, F:, C:, R:, or B:, that are not associated with any of the answers (i.e., lead off Group 4 without answer tags) will be done unconditionally.

Another option uses the RELATED command. We have assumed that RELATED was OFF. By turning RELATED ON (i.e., 0 RELATED ON) in Group 3, the successive tags used in repeats of the same frame depend on which answers were given before, not simply the number of repetitions through that frame. For example, with RELATED ON, the second "A" tag would be used only if the first "A" tag had already been used.

Let's go on to another example frame.

```
P/Q/M/D/C.
*Q

FRAME 3.00 LABEL=*PRES

2. TEXT.
*WHO WAS THE FIRST PRESIDENT OF THE USA?
*

3. ANSWERS.
*0 PHONETIC ON
*0 KEYWORD ON
*A+GEORGE WASHINGTON
*B ABE LINCOLN
*C+G. WASHINGTON
*

4. ACTIONS.
*A F: B:SOMEPLAC
*B R:HE WASN'T THE FIRST, TRY AGAIN... C:COUNT=COUNT+1
*C R:SPELL HIS FIRST NAME.
*
```

Explanation of Frame 3:

The LD labels it PRES in Group 1 and inputs text into Group 2.

3. ANSWERS. Two different correct answers are designated above by the letters A and C. This is perfectly acceptable to PLANIT. However, when C: is used, the correct answer printed out will always be the first one with the plus sign. This group also illustrates the use of the Phonetic and Keyword matchers. The Phonetic matcher encodes all answers into their phonetic equivalent. The Keyword function looks for the answer or answers designated by the LD anywhere in the student's response. Both Phonetic and Keyword are turned on as shown in Group 3. The zero in front of PHONETIC ON and KEYWORD ON tells PLANIT to perform this function before any answers are matched. Their combined use would cause PLANIT to accept the following answer as correct: I THINK IT WAS JEORGE WASHINGTUN. KEYWORD does not accept the answer in reverse order; i.e., WASHINGTON GEORGE would not be accepted. However, if FORMULAS was turned on too, then either order would be accepted.

Note: When KEYWORD is used on a set of answers that contains variations on the one correct (or incorrect) answer, caution should be used to insure proper credit, e.g., A WASHINGTON B GEORGE WASHINGTON

In this example, if KEYWORD is on, Answer B would never be matched unless it appeared first (ahead of A) in the answer set.

The next few frames illustrate the use of CALC statements (i.e., algebraic expressions and primitive control words) in the frames as part of the answer set in Group 3 or following a C: in Group 4. The reader should see the discussion in the section on THE CALC MODE and also the examples in Appendix A.

```
P/Q/M/D/C.
*Q

FRAME 4.00 LABEL =*CALCUSE

2. TEXT.
*WHAT IS THE FACTORIAL OF 35? (USE CALC TO DO THIS FOR YOU).
*

3. ANSWERS.
*1+FACT(35)
*

4. ACTIONS.
*1 F: B:10
*-R:YOU MUST HAVE DONE SOMETHING WRONG, TRY AGAIN...
*
```

Explanation of Frame 4:

3. ANSWERS. In this case the LD is using the primitive FACT (Factorial) as the calculated answer. The important difference is this: All answers tagged with letters (as in the two previous frames) tell PLANIT to look for a literal (or equivalent as in the case of a formula) match between the student's response and the letters in the answer. If the answer is tagged with a number (1-9), then PLANIT assumes that which follows is an expression or function which must be first evaluated and then compared with the student's response. In this manner the LD can define functions in CALC and have them evaluated or operated on during the running of the lesson.

Note: If all answer tags in group 3 are numbers, PLANIT will not allow a response containing letters to be given. Instead, the student will be instructed, NUMERICAL ANSWER PLEASE.

4. ACTIONS. This is the same as in previous frames except that numbers are used here in the same way as letters were to designate actions.

```
P/Q/M/D/C.
*Q

FRAME 5.00 LABEL=*COMP

2. TEXT
*GIVE ME THE MEAN OF COLUMN ONE.
*

3. ANSWERS.
*0 FUNCTION MEAN(X)=(SUM VALUES(J,X) FOR (J=1,N(X)))/N(X)
*1+MEAN(1)
*2 MEAN(2)
*

4. ACTIONS.
*1 F:FINE B:COL2
*2 R:YOU JUST GAVE ME THE MEAN OF COLUMN TWO...O.K., NOW COLUMN ONE.
*1 F:NOW YOU'VE GOT IT.  B:VAR1
*-R:YOU MUST HAVE DONE SOMETHING WRONG, TRY AGAIN...
*
```

Explanation of Frame 5

This frame illustrates the dynamic use of the CALC mode during the execution of the lesson. Essentially, all answers prefixed by a number instead of a letter (in Group 3) put PLANIT in the CALC mode. The LD can then have any expression evaluated at this point that can be done in CALC. The user should have a knowledge of the CALC mode before proceeding further in this frame.

The number 0 differs from the numbers 1-9 in that CALC statements prefixed by the 0 are done before the student answers, while all other numbered CALC statements are done in the order in which they appear from top to bottom until a match is found. Thus no match is attempted on 0 tagged answers.

In this frame we assume that the student has been given a sample of data which appears as two columns of numbers. The sample values also reside in the matrix VALUES (see CALC 5.1.1).

3. ANSWERS. In this group, the ID has defined the function MEAN whose argument is X. He then uses it to operate on the sample data in the matrix VALUES to determine whether the student correctly calculated the mean based on the random sample he just received.

4. ACTIONS. If the student enters the mean of column one correctly the first time, PLANIT responds, FINE, followed by a branch to the frame labeled COL2. If the student answers correctly after a first unsuccessful try, the response will be: NOW YOU'VE GOT IT., followed by a branch to the frame labeled VAR1. If the student enters the mean of column two by mistake, the response will be: YOU JUST GAVE ME THE MEAN OF COLUMN TWO...OK., NOW COLUMN ONE. PLANIT will wait for another answer. If the student entered neither the mean of column one nor column two, the program would print out: YOU MUST HAVE DONE SOMETHING WRONG, TRY AGAIN... and wait for another answer. Numbered and lettered answers both can be used in any group that allows either one. They need not be sequential, numerically or alphabetically ordered (i.e., the actions as a function of answers need not be inserted in the order of their numerical or alphabetical designators). If only numbers are used as prefixes for answers, then PLANIT acts accordingly by telling the student "NUMERICAL ANSWER PLEASE" if he types in letters for his answer.

The next frame illustrates the use of the functions WITHIN and WAIT.

```
P/Q/M/D/C
*Q

*FRAME  6.00 LABEL=*

2.  TEXT.
*WHAT IS THE SQUARE ROOT OF 45 TO TWO DECIMAL PLACES?
*

3.  ANSWERS.
*0 WAIT 45
*1 WITHIN(0.005)
*1+SQRT(45)
*2 SQRT(45) WITHIN (0.05)
*
```

```
4. ACTIONS.
*1 F:FINE
*2 R:YOU'RE CLOSE BUT NOT CLOSE ENOUGH, TRY AGAIN.
*' R:YOU CAN USE THE 'SQRT' FUNCTION IN THE CALC MODE
*' F:SORRY, TIME'S UP.$C: B:HELP
*
```

Explanation of Frame 6:

3. ANSWERS.
The LD set WITHIN (0.005). This means that the student's answer must match the answers in this frame (numerical answers only) within the 0.005 interval. Notice the LD specified different tolerance levels for answers 1 and 2.

The LD also set a time limit of 45 seconds on the student's answer. The actions to be taken if the student does not answer within 45 seconds are prefaced by the prime (apostrophe) in Group 4.

4. ACTIONS. The actions for Group 4 are interpreted as follows:
If the student's answer corresponded to Answer 1, then print out "FINE" and go on to the next frame.

If the student's answer corresponded to Answer 2, print out "YOU'RE CLOSE BUT NOT CLOSE ENOUGH, TRY AGAIN"...then wait for another answer.

If the student did not answer in 45 seconds and this is the first time through the frame, print out "YOU CAN USE THE 'SQRT' FUNCTION IN THE CALC MODE"...then wait for an answer.

If the student did not answer in 45 seconds and this is the second time through the frame, print out "SORRY, TIME'S UP.", followed by the correct answer on the same line. This is accomplished by the use of the dollar sign just in front of the command C:. The dollar sign with no blank after it prevents a carriage return.

Note: The precision of the correct answer that is printed will be determined by the WITHIN parameter.

More examples:

```
3. ANSWERS
*0 WAIT 25
*A+ THOMAS EDISON
*B ALEXANDER BELL
*
```

```
4. ACTIONS.
*A F:
*B R:NO, HE INVENTED THE TELEPHONE, TRY AGAIN.
*-C:
*' R:HINT 1, HE ALSO INVENTED THE PHONOGRAPH.
*' R:HINT 2, HIS INITIALS ARE, T. E.
*' F:TIME'S UP.  C:
```

The WAIT call employed in Group 3 means wait 25 seconds. If minutes are desired, follow the number with the word "MINUTES", e.g., 0 WAIT 25 MINUTES. When the wait time is exceeded (in this case 25 seconds) the commands preceded by primes (apostrophe) are performed. In the example given, the first time the student waits more than 25 seconds without responding, HINT 1 comes out and the timing is reset. The second time HINT 2 comes out, and so on.

The dollar sign "$" can be employed in Group 2 as well as in action command messages to allow the LD to prevent a carriage return between LD inserted commands which were meant to provide successive uninterrupted feedback messages.

The use of the dollar sign in Group 4:

```
3. ANSWERS.
*A+ THOMAS EDISON
*1 PHONETIC ON
*B THOMAS EDISON

4. ACTIONS.
*A F:
*B F:YOU GOT IT RIGHT, BUT NOTE THE SPELLING. $C:
```

If the student's answer associated with B (matched phonetically), he would have seen:

```
YOU GOT IT RIGHT, BUT NOTE THE SPELLING.  THE CORRECT ANSWER IS:  THOMAS EDISON.
```

The absence of the dollar sign would have produced:
YOU GOT IT RIGHT, BUT NOTE THE SPELLING. (followed by a return of the carriage and then)
THE CORRECT ANSWER IS: THOMAS EDISON.

Note: The $ must appear as the last part of a message (no blanks following it); otherwise it will be interpreted as an ordinary character and printed out as part of the message.

The $ can also be used to accomplish automatic text reformatting. See Appendix B for further details.

PHONETIC, KEYWORD, FORMULAS, WITHIN, WAIT and RELATED are automatically turned off after leaving the frame unless entered with the SET command.

E.g.:

```
3. ANSWERS.
*0 KEYWORD ON (Keyword is in effect only for the frame in which this statement
appears.)
```

```
3. ANSWERS.
*0 SET KEYWORD ON (Stays on until a different KEYWORD action is specified, e.g.,
"KEYWORD OFF".)
```

Similarly, PHONETIC ON and SET PHONETIC ON, FORMULAS ON and SET FORMULAS ON, WAIT 20 and SET WAIT 20, WITHIN 0.5 and SET WITHIN 0.5, RELATED ON and SET RELATED ON.

Note: WITHIN can be used as part of a function call or as a part of the function statement.

E.g.:

```
3. ANSWERS.
*0 SET WITHIN 0.5
*1 FUN (3)
*2 FUN (3) WITHIN 0.6
*3 FUN (3) WITHIN 0.8
```

Notice that for 2 and 3 WITHIN was used as part of the answer specification, replacing the SET WITHIN 0.5 if, and only if, the student's answer lies outside the .5 interval about Answer 1. The SET command cannot be used as part of an answer.

E.g.:

```
3. ANSWERS.
*0 SET WITHIN 0.5
*1 FUN (3)
*2 FUN (3) SET WITHIN 0.6
*3 FUN (3) SET WITHIN 0.7
```

In the example just given, the specifications for Answers 2 and 3 would be acted upon as though no SET command appears.

WITHIN can also be turned on automatically in a function call by defining its use in the function definition.

```
3. ANSWERS.
*0 FUNCTION S(X) = 1/X WITHIN 0.5
*1  S(0.5)
*2  S(0.7)
```

Answers 1 and 2 automatically turn on WITHIN. One can also vary the WITHIN interval by defining the function to contain as one of its arguments the tolerance level, e.g.:

```
FUNCTION FN(X,Y) = 1/X WITHIN Y
```

With KEYWORD ON, the student's answer can be embedded in a sentence but only in the same order as in the anticipated answer.

E.g.: "WAS IT THOMAS ALVA EDISON" is a correct answer to: A+THOMAS EDISON, but "EDISON THOMAS" is incorrect.

With FORMULAS as well as KEYWORD turned on, order is not important. EDISON THOMAS would be correct. Periods, commas, question marks, and extra blanks are ignored in all KEYWORD matches.

FORMULAS must be turned on for equivalent algebraic match.

All actions associated with unanticipated answers must be prefixed with a minus sign. Therefore, all commands associate with some symbol. This allows for more than one line of action commands. For example:

```
3. ANSWERS.
*A+ THOMAS EDISON
*B ALEXANDER BELL
*

4. ACTIONS.
*A F:THAT IS CORRECT. THIS IS AN EXAMPLE TO ILLUSTRATE
*F:THE CAPABILITY OF CONTINUING ACTION COMMANDS
*F:ASSOCIATED WITH THE SAME ANSWER DESIGNATOR (IN THIS CASE A)
*F:ONTO MORE THAN ONE LINE. B:NEXT.
*B F:NO HE DID NOT INVENT THE ELECTRIC LIGHT; RATHER,
* R:HE INVENTED THE TELEPHONE, TRY AGAIN.
* -R:YOU ARE WAY OFF, TRY AGAIN.
```

One exception: Commands not prefaced occurring at the beginning of Group 4 are performed automatically independent of the student's response. E.g.:

```
4. ACTIONS.
*F:THIS IS AN EXAMPLE OF THE USE OF COMMANDS NOT PREFACED
*F:OCCURRING AT THE BEGINNING OF THE GROUP. THESE COMMANDS
*F:ARE DONE FIRST. THE REST OF THE GROUP IS PROCESSED
*F:NORMALLY
*A F:THIS MESSAGE FOR ANSWER A  B:5
*B F:THIS MESSAGE FOR ANSWER B  B:5
*- F:THIS MESSAGE FOR UNANTICIPATED ANS  B:5
```

SPECIFICATIONS FOR FORMULAS, KEYWORD, AND PHONETIC*

- What they do:

FORMULAS OFF implies that the student's answer is to be regarded as a standard English reply (i.e., word, number, phrase or sentence).

FORMULAS ON implies that the student's answer is to be regarded as an algebraic formula where commutative, associative, and distributive rules hold. Names are interpreted as parameters of the formula to which arbitrary numbers could be assigned.

KEYWORD ON provides a means for disregarding everything in the student's answer except that which occurs in the LD answer to be matched. The minimal acceptable reply is that which matches the LD's answer. This reply may be a part of a longer reply.

KEYWORD OFF implies an exact match word-for-word.

PHONETIC ON implies that all words in both the LD's answer and the student's reply will be reduced to a phonetic encoding and the encoded messages will be compared - not the original messages.

PHONETIC OFF disallows that function.

- How to use them:

Each of the three is controlled in exactly the same way. They can be turned on or off in CALC or in any statement on which CALC operates. Once turned on, they will be automatically turned off again when advancing from one frame to the next unless the SET operator is included. If they are SET on, they will remain on until that status is changed by a subsequent declaration of the status of that function.

The chart below shows several uses:

	In CALC	In Answer Group	In Action Groups
Temporarily ON	KEYWORD ON	0 KEYWORD ON or 1 KEYWORD ON	C: KEYWORD ON
Set ON	SET KEYWORD ON	0 SET KEYWORD ON or 1 SET KEYWORD ON	C: SET KEYWORD ON
OFF	KEYWORD OFF	0 KEYWORD OFF or 1 KEYWORD OFF	C: KEYWORD OFF

Note: In the answer group, the 0 implies that the action will be taken before the student responds, while 1 or greater implies that the action will be taken at the position found in the process of answer-matching.

---

*For details on techniques used in FORMULAS and PHONETIC, see Appendices C and D respectively.

- How they work:

Their function will be noticed only in question-type frames. The following chart shows the various combinations.

	FORMULAS ON	FORMULAS OFF
KEYWORD* ON (letter tag)	Answers will be treated as algebraic formulas. To match, at least the keywords must be found... in any order	Words appearing in the answer must appear somewhere, properly ordered, in the student's reply.
KEYWORD OFF (letter tag)	Answers will be treated as algebraic formulas. Every word in the answer must occur in the reply and no others... in any order.	Everything in the answer must appear in the same way in the student's reply. Only leading and trailing blanks are ignored.
NUMBER TAG (Keyword irrelevant)	Student's answer can be an arithmetic expression with numbers and arithmetic operators only (no letters)	Student's answer can only be a number (in simple or scientific form, e.g., .1234*10**5).

In each case, PHONETIC ON simply phonetically encodes the words in the LD's answer and student reply before matching is attempted.

- Operational definitions:

A word is a string of letters set apart with blanks, i.e., EDISON.

With FORMULAS OFF, any string of characters set apart by blanks are handled as unit "words."

With FORMULAS ON, the word is a string of letters that may or may not have digits attached on the right, i.e., MEAN SUM1 RHO12

- Restrictions and special conditions for answers tagged with letters:

---

*Periods, commas, question marks, and extra blanks are ignored with KEYWORD on.

1. With **FORMULAS ON**, punctuation is ignored and arithmetic symbols retain their function.

2. With **FORMULAS OFF**, no symbol has special meaning and must be matched exactly. Punctuation is not ignored unless KEYWORD is on.

3. FORMULAS ON is often useful for other than formula replies, e.g., THOMAS EDISON would allow EDISON, THOMAS to match correctly (the comma is ignored and order is not observed).

4. All answers are restricted to a single line.

(Q) THE DICHOTOMOUS FRAME (POS/NEG)

This frame is considered a variation of the Q-type frame. The main difference is in Groups 3, 4, and 5 (the regular Q frame does not have Group 5). The following example illustrates the building of a POS/NEG frame in the verbose mode.

```
(P)ROBLEM/(Q)UESTION/(M)ULTIPLE CHOICE/(D)ECISION/(C)OPY.
*Q

FRAME 6.00 LABEL=*DIFF

2. SPECIFY QUESTION
* USING THE SAMPLE PREVIOUSLY GENERATED, IS THE MEAN OF EITHER
* COLUMN GREATER THAN THE NUMBER OF MINUTES THAT HAVE ELAPSED
* SINCE MIDNIGHT?
*

3. SPECIFY ANSWERS
*POS/NEG

4. SPECIFY CONDITIONS FOR POS.
* IF MEAN(1)  GR TIME OR MEAN(2) GR TIME
*

5. SPECIFY ACTIONS
* +F:  B:GOOD
* -B:NG
```

Explanation of Frame 6:

3. ANSWERS. The presence of two words separated by a slash (/) indicates to PLANIT that this is a POS/NEG type frame, e.g., RIGHT/WRONG, INSIDE/OUTSIDE. POS/NEG means that any positive statement would be accepted as a correct answer providing the conditions specified in Group 4 are true, and that if the conditions are not true, then any negative answer is correct.

PLANIT maintains a list of positive and negative answers like: YES, RIGHT, TRUE or NO, INCORRECT, etc. Any answer of this form that PLANIT finds in its list will be matched against the student's answer. If a match if found, PLANIT will act accordingly. If no match is found, PLANIT will print out: SORRY, THAT ANSWER WAS NOT ANTICIPATED, PLEASE REPHRASE IT OR USE A SYNONYM.

POS/NEG is special in that any positive or negative answer is considered. Had the LD used INSIDE/OUTSIDE, for example, only INSIDE or OUTSIDE would be the anticipated answers. However, the match will be as in KEYWORD where the correct answer can be embedded in an answer of more than one word. Negations are also handled, e.g., NOT INSIDE. Words ending in 'T are handled as negations.

4. CONDITIONS. In this group the LD specifies the conditions under which any affirmative answer would be correct. The conditions will be in the form of two mathematical expressions joined by a relational. The truth of the relation will be tested at the time this group is encountered and the appropriate actions in Group 5 will follow.

Note: The relational operators used in specifying the conditions are: LS - less than; LQ - less than or equal to; EQ - equal to; GQ - greater than or equal to; GR - greater than; and NQ - not equal to.

The omission of this group implies that the first listed answer is unconditionally correct, e.g., if YES/NO was given, YES would be treated as correct.

5. ACTIONS. Action commands are the same as in the regular question frame except that the plus (+) is used to indicate which action should take place if the student answers correctly. Lines containing actions prefixed by a minus (-) will be done if the student is wrong.

(M) THE MULTIPLE CHOICE FRAME

This frame is constructed of groups identical to those of the Q-type frame in that no dichotomous answer variation is permitted. All examples of the regular Q-frames apply for Groups 2, 3, and 4. The main difference between the Q and M frames becomes apparent in the EXECUTION mode. Since the frame is designated as M-type, PLANIT will actually print out the letter-tagged answers specified in Group 3 with their associated tags. The plus sign (+) will not be printed if next to an answer tag. If an answer is tagged with a number, that line will not be printed, e.g., 0 WAIT 10.

```
P/Q/M/D/C
*M

FRAME 8.00 LABEL=*MULT
```

```
2. TEXT.
*USING PI FOR 3.14159 AND R FOR THE RADIUS, WHAT IS THE FORMULA FOR THE AREA
*OF A CIRCLE?
*

3. ANSWERS.
*A+PI+R**2
*B (4/3)*PI*R**3
*C PI*2*R
*

4. ACTIONS.
*A F:  B:OUT
*-BC R:
*
```

Explanation of Frame 8:

The LD labels it MULT.

During the EXECUTION mode, the answers would come out as choices for the student.

E.g.:

```
USING PI FOR 3.14159 AND R FOR THE RADIUS, WHAT IS THE FORMULA FOR THE AREA OF
A CIRCLE?

A PI*R**2
B (4/3)*PI*R**3
C PI*2*R
*
```

If the student should respond with a statement or chooses one of the letter tags that do not appear, PLANIT will automatically respond with:

CHOOSE ONE OF THE ABOVE LETTERS.

4. ACTIONS. If the answer is A, the program will print out an affirmative answer (selected randomly) and then branch to the frame labeled OUT. For Answer B or C, PLANIT will respond with WRONG, TRY AGAIN and wait for another answer.

(D) THE DECISION FRAME

This is an extremely powerful frame and can be used for determining just about anything the student has done. In this frame the LD specifies any of the three commands (F: C: B:) as a function of student performance over several frames. He can also query the contents of items set in CALC. Decisions may be logically connected, as will be seen later.

Decision frame questions are composed of the following primitives:

1. The Connectives: IF, AND, OR.
2. The Relational Operators: LS - less than; LQ - less than or equal to; EQ-equal to; GQ - greater than or equal to; GR - greater than; and NQ - not equal to.
3. The Conditions: RIGHT, WRONG, SEEN, USED, MINUTES, +, -.
4. Frame numbers and labels.
5. Control words: FROM, ELSE, END, ALL, NONE.
6. Lettered or Numbered answers.
7. CALC expressions.

The beginning of any decision question must start off with the "IF" connective. The question can run over onto more than one line, but all lines must start off with a connective.

The questions can have any one of three forms:

A. IF Frame number (or label), answer tags (letters, numbers or symbols).
E.g., IF 5,AB 7-12,+ 3,1P 6,+-
This is interpreted as follows: If the student went through Frame 5, Answer A or B and then went through Frames 7 to 12 (inclusive) correctly and then Frame 3, Answer 1 or P and then saw Frame 6. The first form is the "Pattern" form for it describes an exact pattern. In order for this question to be satisfied, the student must go through the frames mentioned in the exact order specified with no deviations to frames in between nor to the answers given.

Note: IF 2,- means IF 2-2,- i.e., the form of this IF statement is always IF A-B,- etc.; if no B then PLANIT assumes B = A. This may or may not present problems, for example IF 2,- would be true if the student went through Frame 2 twice in a row, wrong each time. However, the statement would be false if going through Frame 2 twice in a row he got it right at least once. The same holds true for three times in a row and so on.

One should be aware about the dangers of inserting frames between any frames that already appear consecutively in an IF Statement of Type A, that would effectively nullify the statement unless it too were "edited."

B. IF Relational number RIGHT (WRONG, SEEN, USED, MINUTES) Frame numbers (labels).

E.g., IF GQ 5 WRONG 1,7,LAB-25
If the student got at least five wrong out of Frames 1, 7, and all frames between the frame labeled "LAB" and Frame 25.

E.g., IF RIGHT 3-5. If he got Frames 3 to 5 right.
E.g., IF EQ 5 WRONG. If he got five frames wrong.
In this form any part is optional and can be left out except RIGHT, WRONG, SEEN, USED OR MINUTES.

E.g., IF LQ 20 MINUTES 1-5,7. If the student went through Frames 1-5 and 7 in at most 20 minutes. This includes all repetitions of frames.

E.g., IF FACT USED 1, 2 AND MEAN USED 3.
If he used the primitive CALC function "FACT" in either of Frames 1 or 2 and used the LD defined function "MEAN" in Frame 3.

C. IF CALC Expression Relational CALC Expression
E.g., IF CNT GQ 5. If the contents of CNT is greater than or equal to five.

E.g., IF FACT(CNT)-SQRT(STUDIQ(5,CNT))*TAN(50)**2 LS COMB(5.CNT).
If the factorial of the contents of CNT minus the square root of the element of the matrix STUDIQ whose subscripts are 5 and the contents of CNT, times the tangent squared of 50 is less than the combination of 5 things taken CNT at a time.

All of the three forms, A, B, and C can be connected by any of the connectives AND, or OR.

Grouping notations (ELSE, END). More involved decision statements can be written by using the control words ELSE or END. The use of ELSE has a double implied branch, e.g.,

```
(1) IF FROM 30 GQ 5 RIGHT 35-45
(2) F:FINE NOW YOU ARE GETTING THE RIGHT
(3) F:IDEA, LET'S GO INTO IT A BIT MORE.
(4) ELSE F:LET'S LOOK AT IT AGAIN B:35
(5) IF FROM 30 GQ 7 RIGHT 35-45 B:50
(6) ELSE B:46
```

In the above example: If the question on line (1) is true, then lines 2 and 3 are performed. Control then continues on line 5. If line (1) is not true, control immediately skips to the ELSE conditions on line 4. Note: If there was no Branch on line 4 (B:35), then control would go on to line five. The two implied branches are as follows: If line one is true, the branch is line 3 to line 5; if line one is false, the branch is line 1 to line 4. The implied branching is different when using END, e.g.,

```
(1) C: COUNT=0
(2) IF 5,A 7,+
(3) F:FINE NOW YOU'VE GOT IT.
(4) C: COUNT=COUNT+1
(5) END C :COUNT=COUNT+1
```

In this second example note COUNT is first cleared to zero. If line 2 is true, lines 3 and 4 are performed and control then falls through the END and continues on line 5. If line 2 is false, control goes right to line 5. The implied branch is only when the statement is false. In either case control continues right through the END unless previously directed via a "B:" on line 4.

Notice: If line 2 is true, COUNT is equal to 2 when at line 6. IF line 2 is false, COUNT is equal to 1 when at line 6.

E.g., IF 3,AB LAB,+ OR 2 SEEN LOGIC AND IQ LQ 120.
OR GQ 3 RIGHT HIST-MTH.

If the student went through Frame 3, Answer A (or B) and then was correct on LAB OR he has seen the frame labeled LOGIC twice AND the contents of the item, IQ, is less than or equal to 120 OR he got at least three right out of all the frames between HIST and MTH (inclusive).

The terms of these expressions are separated by OR's. Although no parentheses are permitted, they will be used here to clarify the order in which the example just given would be evaluated.

IF (3,AB LAB,+), OR (2SEEN LOGIC AND IQ LQ 120)
OR (GQ 3 RIGHT HIST-MTH).
The terms are always evaluated left to right.

Now note the next example:

```
P/Q/M/D/C
*D

FRAME 9.00 LABEL=*DET



2. CRITERIA.
* IF 3,- F: LET'S TRY SOME EXAMPLES B:EXAM
* IF 5-10,SK EXAM,+ OR LQ 2 SEEN 20,23-25 B:72
* IF 2 RIGHT HIST-MTH AND STUDIQ LQ 130 OR 7-10,ABCD56
* AND ALL SEEN 5-35 C:CNT=10 C:LINK(2)=25 B:105
*
```

Explanation of Frame 9:

The first line reads: If he got Frame 3 wrong, printout the feedback message "LETS TRY SOME EXAMPLES" then branch to frame whose label is EXAM.

The second line reads: If he went through Frames 5 to 10, answers S or K, and then went through the frame whose label is EXAM correctly OR he has seen at most two out of Frames 20 and 23 to 25, then branch to Frame 72.

The third line will be easier to follow if we label its terms.

Let: 2 RIGHT HIST-MTH→A; STUDIQ LQ 130→B: 7-10,ABCD56→C;
ALL SEEN 5-35→D
Then line three is read as: IF (A AND B) OR (C AND D).
Then set CNT equal to 10, LINK (2) equal 25 and branch to FRAME 105.
E.g.,

```
*IF  NONE SEEN 5-50 OR 5,A 6,+
*F:FINE NOW YOU'VE GOT IT  C:  TEMX=FACT(40)  B:67
```

Finally, those last two examples introduce the use of ALL and NONE. In the first one we have: If the student has seen ALL the Frames 5-35. In the second we have: If he hasn't seen any of the Frames 5-50.

As shown in Group 4 of the Q-frame, the action commands can be grouped with a particular answer. Similarly in the Decision frame, action commands F:, C: and B: (R: not used) can be grouped following an IF statement. Remember, to continue a feedback message onto more than one line, the lines must start with "F:". As in the last Group 4, example given on p.25, commands not prefaced occurring at the top of the frame are done first.

E.g.,

```
FRAME X LABEL=X
2. CRITERIA.
*F:THIS IS DONE FIRST, THE REST OF THE GROUP
F:
*IF  1,A...etc.
```

Search Boundaries: (FROM)

All examples shown thus far imply restriction to the scope of the search. If this is the first time this frame is used (i.e., this frame number appears nowhere else in the student's record), the search begins from the first frame as before. If, however, this decision frame was used before, then the search starts off from the frame record associated with the last time this decision frame was used.

The LD can override the limitation by specifying the frame number from which the search will begin. He can use the primitive "FROM" followed by the frame number or label. The primitive "FROM" is used just after the "IF". E.g., IF FROM 20, GQ 5 RIGHT...etc., or IF FROM EXAM, GQ 5 RIGHT...or IF FROM 5,6, A 7, BCD...the search starts off from the last occurrence of that frame record whose number is the same as that which follows the primitive "FROM" in the decision statement.

If the frame number is not found, a search is made of the entire record to see if the pattern is satisfied anywhere.

Remember, the pattern must exist exactly as specified in the decision question (with no deviations). The search starts from the beginning of the lesson and ends when the question is satisfied or until the last record is examined, whichever comes first.

Finally, for RIGHT and WRONG, D (decision) and P (problem) type frames are ignored as well as those frames for which no right answer was identified (i.e. no "+" sign).

Consider the following examples:

	FRAME	ANSWER	RIGHT/WRONG	FRAME TYPE
1.	1	A	+	Q
2.	2	K	-	Q
3.	5	A	-	M
4.	7	C	+	Q
5.	11	C	+	Q
6.	4	P	-	M
7.	5	B	+	M
8.	7	C	+	Q
9.	7	C	+	Q
10.	9	C	+	Q
11.	10	C	+	Q
12.	20	O	o	D
13.	15	S	+	M
14.	35	O	o	P
15.	12	O	o	D
16.	20	O	o	D

(Frame 20) Figure 2

A) IF FROM 1 5, AB 7-10, C
B) IF 4 SEEN 5, 12-35
C) IF 3 RIGHT 5, 12-35

Referring to Figure 2, Question A would be satisfied. Note that the pattern is disturbed in Lines 3-5, but the search continues and a complete match is found in Lines 7-11*. By removing the "FROM", Question A would not be satisfied, since the specified pattern occurred prior to the last use of this decision frame (Frame 20 in the example).

Question B would not be satisfied. Although the student did see four frames, he did not see all four in between the limits as determined by the use of Frame 20 previously.

Question C would not be satisfied for similar reasons given to B; i.e., he was correct on them all, but not within the two appearances of Frame 20. Notice that by adding "FROM 2" to Questions B and C, both would then be satisfied, i.e.,

```
B)      IF FROM 2 4 SEEN 5, 12-35
C)      IF FROM 2 3 RIGHT 5, 12-35
```

The answer column will always be zero (lines 14-16) if the frame has no group 3. If group 3 appears in a frame, then either a letter or a minus sign appears in the answer column, whether or not the frame has a group 4.

(P) THE PROBLEM FRAME

The problem frame should be viewed as an environment generator. Generally speaking, this frame sets up the parameters for a series of operations. All information in this frame remains in effect until another problem frame is executed, and then, only to the extent of the changes made to the groups in the next problem frame. The frame essentially does two things: (1) sets up the extent of aid the student can get; and (2) provides for a random sample to be generated during the execution of the lesson.

Groups 2-5 set up aids for the student. Groups 6-9 set up the parameters used in generating a sample.

The Problem frame would most likely be embedded between some Q or M type frames. The Q frame preceding this frame could prepare the student by telling him that a sample will be generated upon which he would be expected to perform some computations, e.g., calculate the mean, variance, etc. The frame following this P-frame (be it Q or M) would then ask him to enter his answers.

---

*All frames in between 7 and 10 need not occur for a match. The only requirement is that the frame numbers are sequential (i.e., the next number is greater than or equal to the preceding one) and that the two frames mentioned (7 and 10) appear. This is with respect to pattern-type questions and does not apply to forms using RIGHT, WRONG, ETC.

Consider the following example (at the verbose level):

```
(P)ROBLEM/(Q)UESTION/(M)ULTIPLE CHOICE/(D)ECISION/(C)OPY.
*PROBLEM

FRAME 11.00  LABEL=*BASIC

3.  OPTION CONTROL- ALL/ALL BUT(FACT, TAN)/NONE/ONLY(SQRT)/
ADD(SIN, COMB)/NOT(COS, COT).
*ONLY(LN),ADD(SEC,CSC,FACT).
*

5.  LIST STEPS (IN ORDER) FOR STEPS TO THE SOLUTION.  (E.G., FIRST COMPUTE
THE MEAN: NOW COMPUTE THE VARIANCE.)

*TO COMPUTE THE MEAN, SUM UP THE VALUES IN THE SAMPLE AND THEN
*DIVIDE BY THE NUMBER OF THEM;
*THE FORMULA FOR THE MEAN IS:  M=(SUM DATA(I,1)FOR (I=1,N))/N
*WHERE N IS THE NUMBER OF ELEMENTS IN THE SAMPLE.
*

6.  SPECIFY SAMPLE SPECIFICATIONS- (O)NE GROUP/(T)WO INDEPENDENT
GROUPS/(M)ATCHED GROUPS/ALSO SPECIFY STUDENT PREFERENCE (Y/N).
*ONE  N

7.  SPECIFY DISTRIBUTION-  N/R/B.
*N

8.  SPECIFY-   MEAN RANGE SKEW(N/R/L)  SIZE.
*               60   26      N         20

9.  SPECIFY SAMPLE HEADING.
*NUMBER   BOYS HEIGHT IN INCHES
*  X       X
```

Explanation of frame 11

The LD typed in "PROBLEM" after the program requested the frame type. He could have typed in "P" and gotten the same results. PLANIT responds with frame number and requests a label. The LD typed in BASIC following the asterisk. The label "BASIC" is now connected with frame eleven and can be referred to either by its label or number.

Groups 2 and 4 of the problem frame are presently not being used.

3. OPTION CONTROL. This allows the LD to specify what options of PLANIT will be available to the student (during the course presentation) for that problem. In the verbose mode, PLANIT types out some examples as shown. The following formats are acceptable: ALL - all options are available; ALL BUT (J,K,M-N)-only the listed options cannot be used; NONE-no options can be used by the student; ONLY (J,K,M-N)- just options J and K and options M through N can be used; ADD (J,K,M-N)- add options J, K and M through N to those already allowed; NOT (J,K) - all options that were allowed before except J and K. (Options listed without a modifier are assumed to be ADDed).

At the time of this writing there exist 21 options, i.e., prebuilt routines which are part of the program. The use of these options by the student is under the control of the LD. The options are: STEPS HELP RANK SQRT COMB FACT LOG LN SIN COS TAN COT SEC CSC ABSOLUTE TRUNCATE ZTOP PTOZ PTOT PTOX INVERT (see CALC 10.1.5). The LD allowed options natural log, secant and cosecant. Initially all primitives are allowed. The options allowed (or not allowed) in this group stay in effect until the next problem frame is executed (unless the LD does not use Group 3 in that frame).

While the student is taking the lesson, he may want to enter CALC and use some of the options mentioned above. In the example given, he could use: LN, SEC,CSC and FACT; e.g., he would type:
*FACT(5) and PLANIT responds with:
120.0 (the asterisk is typed by PLANIT).

If he tried to use SIN(45), PLANIT would not recognize SIN, since the LD had not allowed it. All functions that are primitives of PLANIT like FACT, SIN, TAN, etc., or defined by the LD are handled in the same manner, i.e., wherever one can refer to a primitive (e.g., FACT), one can also refer to any function defined by the LD and vice versa.

E.g.: Suppose the LD defined the function, MEAN. Then in Group 3 of the P frame (option control), the LD can "ADD (MEAN)". Similarly in Group 3 of the Q frame, one can ALLOW or PROHIBIT MEAN. Finally, all options and functions are recorded when they are used by the students. When the LD DISPLays the student's records, he will be shown which options the student used on any frame.

5. LIST... This group allows the LD to give hints to the student. The LD can type in comments. Each comment is separated by a semicolon. During the execution of the course, the student (in the CALC mode) can request steps to the solution by typing in STEPS. The first time, he gets the first comment. Typing STEPS again produces the second comment and so on. In the example given, assume we have asked the student to compute the mean of some sample data. The student is stuck and doesn't know how to go about it, so he types in STEPS. PLANIT responds with:

STEP 1.
TO COMPUTE THE MEAN, SUM...THEM.

The student is still stuck and so types in STEPS again. PLANIT responds with:

STEP 2.
THE FORMULA FOR ...SAMPLE. Since the LD has not put in any more comments, PLANIT will respond with ALL STEPS COMPLETED, should the student type STEPS again.

6. SAMPLE SPECIFICATIONS requires the LD to specify the sample structure and student preference. For sample structure, he may choose a single group, two independent groups or two matched groups. In the example he has chosen a single group. On the same line, a "Y" or "N" determines whether or not the student's preference will determine the group size(s).

If a "Y" is entered, the LD should follow it with a message asking the student to input the sample size he wants. In the absence of a message following the "Y", PLANIT will supply the message, SPECIFY SAMPLE SIZE. However in this example, the "N" entry denotes that the sample size is not to be determined by the student.

7. DISTRIBUTION lets the LD specify the probability with which sample data will fall in certain intervals of specified range (range is specified in Group 8 discussed below). The alternatives allowed for the distribution, which may be univariate or bivariate, will be a combination of one or two out or NORMAL, RECTANGULAR, or BINOMIAL and will also be a function of the answer given to 6 above, e.g., one group: DIST (N/R/B); two groups: DIST (NN/NR/NB/RR/RB/BR/BB). In the example, the LD has specified a normal distribution for his single group.

8. PARAMETERS allows the LD a great amount of freedom in specifying the parameters for the hypothetical population from which the sample will be drawn. The responses are to be placed on the next line approximately under the header name (separated by at least one blank). The header will contain only those names that are pertinent to the distribution and sample structure as a function of 6 and 7 above. In this example, the LD has specified "population parameters" comprised of a mean of 60, value range of 26 (± 13 from the mean), no skewing of the distribution, and a sample size of 20. The sample values are generated randomly and transformed into numbers that comply with the parameters in Groups 6, 7, and 8. For example, the following sample might be produced from the above illustrations.

NUMBER	BOYS HEIGHT IN INCHES
1-	57
2-	69
3-	71
4-	60
5-	71
6-	64
7-	68
8-	64
9-	58
10-	47
11-	50
12-	58
13-	68
14-	63
15-	59
16-	54
17-	61
18-	54
19-	63
20-	70

The mean of this sample will be an estimate of the designated mean of 60. The range dictated that no sample value exceeded 73 or was less than 47 (the mean being located at the center of the range). The values are approximately normally distributed as specified with a sample size of 20.

The standard deviation does not explicitly appear in Group 8 but is estimated to be one-sixth of the range.

Suppose you wanted to produce a sample of fifteen IQ scores from a normally distributed population whose mean is 100 and whose standard deviation is 15. Your groups might look like this:

```
6.    SPECIFY SAMPLE SPEC....
      * ONE  N

7.    SPECIFY DISTRIBUTION- N/R/B
      *N

8.    SPECIFY-  MEAN  RANGE  SKEW(N/R/L)  SIZE
                100     90   N              15

9.    SPECIFY SAMPLE HEADING
   *  STUDENT NO.    IQ SCORE
   *        X              X
```

Notice that the standard deviation has been changed to the range by multiplication (15x6=90). The sample values produced by that problem frame will now resemble normally distributed IQ scores.

Other parameters would have been requested in Group 8 if different responses had been given in groups 6 and 7. If you specified two groups, you would also be asked for parameters for the second group (RANGE2, SIZE2, etc.). The MEAN of the second group is determined from the MEANSHIFT or the desired distance between the two means. If matched groups were requested, you would be asked for the correlation coefficient (CORR). If a binomial distribution was requested, you would be asked for the PROB for that group (i.e., the probability governing the appearance of ones in a sample of ones and zeros). Certain parameters, listed below, will be filled in appropriately in random fashion by the computer if an "R" is specified in place of a number.

Responding with R or L to the SKEW request causes the distribution to be skewed either to the right (R) or left (L). It is not a true skew parameter. If a right (R) skew is specified, the upper one-third of the range will be truncated from the sample. If a left (L) skew is specified, the lower one-third of the range will be truncated. This will occur similarly for either the normal or rectangular distributions. If the sampling distribution was rectangular, the range will be reduced by the amount truncated. If the distribution was normal, the sample will be moderately skewed in the direction indicated. In either case, the mean of the sample will be affected considerably by the truncation. It will be much lower than specified for a left (L) skew or higher for a right (R) skew.

Listed below are all the possible names that will occur in Group 8. Following the names are the possible entries that can legally be specified for each. These names will not all appear in every problem frame. Which ones appear will depend on how Groups 6 and 7 are filled in.

MEAN 1	mean of the distribution used in generating Group 1 - real number
RANGE 1	range of the distribution used in generating Group 1 - real and positive (The standard deviation used in generating samples from a Normal distribution is arbitrarily defined as one-sixth of the specified range.)
SKEW 1	(N/R/L) No, Right, Left
MEANSHIFT	constant shift value between means - real number (MEANSHIFT + MEAN 1 = MEAN for the second population to be sampled.)

RANGE 2	same as RANGE 1 but for the second group
SKEW 2	(N/R/L)
SIZE 1	size of first group (2-100)
SIZE 2	same as above
CORR	Pearson Product Moment correlation used in generating samples consisting of two matched groups.
PROB 1 or PROB 2	probability of obtaining a "1" in a binomial sample of ones and zeros
MEANSHIFT, RANGE 2, CORR and PROB 2	will accept an "R" response which will cause the parameter to be filled with a reasonable random number.

9. SAMPLE HEADER requires two lines of information. The first line contains header names which label groups in the sample columns. In the example given, the LD chose: NUMBER BOYS HEIGHT IN INCHES as his header. Had he required a sample consisting of two groups, he might have added a third label, GIRLS...etc. For the second line, the LD types in a character (any character) denoting the position of the sample values, right justified, for each labeled group, and also the number format for the data.

The data will be printed for the student in one or more columns with a current maximum of three. The number of columns will depend on (1) how many were designated in group 6 (either one or two) and (2) whether or not the rows are to be numbered. Normally, the rows will be numbered (see the example under the discussion of group 8). If this column of sequential row numbers is not desired, type "NO" for the first entry in line 2 of group 9.

The remaining entries on line 2 of group 9 will be used to position the data columns and designate the format. The letter "X" is suggested as a column locator, however other characters will also work. The X's should be placed on the line at the column where the lowest order digit of the data is desired. If the data is to be integer, nothing more is required. If, however, you wish to carry digits past the decimal point, indicate this by following the X with a decimal point and as many more X's as digits you want to generate. For example, X will produce integers, X.X will produce one place decimal numbers (e.g. 32.5, 9.0, etc.), X.XX will produce two place decimal numbers (e.g. 7.25, 3.14), etc. If the decimal point is used, all decimal points in that data column will be vertically aligned. To illustrate, the heading might read:

```
G9:   HEADER
*       SUBJECT       BEFORE      AFTER
*         X            X.X         X.X
```

To the student it would appear

```
        SUBJECT       BEFORE      AFTER
          1-           21.4        25.3
          2-           19.5        18.2
```

etc. - - - - - - - - - - - - - - - - -

Again, the heading might read:

```
G9:   HEADER
*       ADD THESE NUMBERS
* NO      X
```

The student would see:

```
        ADD THESE NUMBERS
           429
            36
           219
```

etc. - - - - - - - - - - - - - - - - -

(See Appendix E for additional capabilities of the problem frame.)

(C) THE COPY FRAME

The Copy frame is very easy to use and is more of a building aid than a frame in its own right. The use of this command allows one to copy any other frame already built into the same lesson. After P/Q/M/D/C, one simply types: C N, where N stands for the number or label of any frame already built. To this PLANIT will respond with FRAME DD.DD LABEL=* where DD.DD is the number of the sequential position of this new frame. At this point, Frame N has been copied completely. However, PLANIT will still go through the motions of passing through each group, allowing the LD to change any group he desires. Once any line of a group is changed, that entire group has been replaced. To pass on from one group to another, simply strike the space bar and carriage return. To get out of the frame, just enter a "$".

```
P/Q/M/D/C.

*C 2 (or MATH, Frame 2's label)
```

```
*FRAME 12.00 LABEL = *COP

2. TEXT.
*$
```

P/Q/M/D/C.
Explanation of Frame 12

The LD wanted an exact copy of Frame 2.00 to be the 12th frame of the lesson. The only change made was to give it a different label. Having done this, the LD wants to go on to the next frame and does so by entering a "$" as the first character of the line. This is true for any frame. You can always get out of any frame by entering a "$" as the first character of any line. You can always get out of any group by entering only a blank on any line.

The advantage of the C frame is that it allows the LD to generate similar frames in a given lesson simply by copying a previous frame intact and then changing those parts wherein the LD wishes it to differ.

Note: If no label is desired, the LD must strike the space bar and carriage return. If the LD uses the "$" in Group 1, i.e., the frame header, he will lose Frame 12. In short, in order for PLANIT to retain a frame, it must at least have a label or some data in one of its groups. The Copy frame need not have a label but it must be terminated in Group 2 or later. For example:

```
FRAME 12.00 LABEL=*$
P/Q/M/D/C
 Q
FRAME 12.00 LABEL=*
```

The dollar sign terminated the frame in Group 1; therefore the frame was not saved.

SPECIFICATIONS FOR LINKING LESSONS

Any lesson in PLANIT can use any other lesson built in PLANIT. This can be a very useful device. Suppose the LD is teaching a course on statistics and at a certain point along the way he discovers that the student is weak in elementary set theory. Suppose further that the LD knows of the existence of a well-designed course on set theory (also built in PLANIT). At that point in the lesson the LD can have a branch to the other lesson.

PLANIT provides for 10 linking items that can be passed along from lesson to lesson LINK(1)...LINK(10). In this way lessons can communicate with each other; e.g., LINK(1) could contain the frame number past which the LD does not want the student to go in the called lesson. LINK(10) might contain the frame number for the called lesson to start at, and so on.

Also note that the called lesson might call on another lesson. There is no limit to this nesting.

It is conceivable (given a large enough library of lessons) that a LD could very simply put together a very long course or set of courses by merely calling on the appropriate set of lessons in the library, constructing his frames in such a manner that they look like nothing more than a table of contents, or perhaps, a semester's outline.

e.g.,

```
P/Q/M/D/C.
*M
FRAME 13.00 LABEL=*SWITCH

2.  TEXT.
*WHAT IS THE DIFFERENCE BETWEEN 5¼ and 4½?
*

3.  ANSWERS.
*A I DON'T KNOW
*B 3/4
*C 12/4
*

4.  ACTIONS.
*AC C:LINK(1) = 1.0  C:LINK(2) = 7.0   B:ARITH
*B F: B:ONWARD
*
```

Explanation of Frame 13.00

Here we see the use of C: followed by a CALC statement. Remember C: by itself prints out the fixed message "THE CORRECT ANSWER IS:" followed by the correct answer. However C: followed by some expression is interpreted by PLANIT as a CALC statement. Thus we can perform a CALC statement as a function of the student's answer (e.g., C: COUNT = COUNT+1).

This frame uses C: to set up linking items for communicating with another lesson that this lesson is about to call on, viz., ARITH. Notice the call is just like a branch to another frame. If PLANIT does not find the label in its own lesson, it assumes this is the name of another lesson and attempts to use it. If the lesson called is not found, PLANIT will print out an error message.

In the example given (Group 4): If the student gave Answer A or C, then LINK(1) is set equal to 1 and LINK(2) is set equal to 7 followed by a call to the lesson whose name is ARITH.

(#P,I,D) FRAME EDITING

Once a frame or set of frames is built, they can be printed, deleted, or inserted; this is also true for parts of frames. The general form for any of these three commands are F1-F2,G1-G2,L1-L2,E (don't forget the commas). Where F1,F2 stand for frame numbers, G1,G2 group numbers, L1,L2 line numbers, and E stands for P,I, or D (Print, Insert, or Delete). F1,F2,L1,L2 can be numbers with fractional parts (e.g.,5.30). G1,G2 must be an integer. For inserting only (I), F2G2L2 cannot be used. (One can only insert a line or frame/per command. Group insertion starts at the group number specified but PLANIT continues passing through the groups until the last group or when the LD enters $ and CR.) Omission of line numbers signify the entire group, omission of group numbers signify the entire frame.

For example:

- To delete line 2, group 2, of frame 3.

```
*3,2,2,D
```

  (Program types out asterisk when done)

- To insert a line of text after line 1, group 2, frame 1:

```
*1,2,1.1,I

*
AMONG OTHER THINGS
```

  NOTE: PLANIT types an asterisk and a CR allowing you to line up your input as desired. This also tells you that you are in the line insert mode.

- To replace line 1.1, group 2, frame 1.

```
*1,2,1.1,I
```

  AMONG OTHER THINGS (the program prints out the requested line, an asterisk, and a CR, you type in the new line starting right under the asterisk in column one. Blanks may be entered to format the line as desired.)

```
*
AMONG OTHER THINGS
```

  NOTE: If you change your mind (i.e., you don't want to replace the line), just type in the next edit command prefaced with a #.

For example:

*1,2,1.1,I	The LD wants to change 1, 2, 1.1
*AMONG OTHER THINGS	Program prints 1, 2, 1.1 and waits for next command.
*#1,3,I	LD changes his mind and requests frame 1, group 3
3. OPTION CONTROL ALL/NONE/ADD(SIN, COMB)/etc.	Program prints out the header for frame 1, group 3
*ALL	LD wants to allow all options

- To print all of frame 84 (note if the LD does not enter P,D, or I, PLANIT assumes P, however no line numbers will be printed out).

```
*84,P

FRAME 84.00 (Q)

G 2.  TEXT
[1.00]
AND WHAT IS THE SUM OF THE SQUARED DIFFERENCES FOR THIS TABLE?

G 3.  ANSWERS
[1.00]
1 SCP
[2.00]
2 SMD
[3.00]
3+SSD

G 4.  ACTIONS
[1.00]
1 R:YOU ARE STILL READING THE FIRST PROBLEM.  TRY SSD THIS TIME.
[2.00]
2 R:SUM OF THE MEAN DIFFERENCES WAS THE LAST PROBLEM.  TRY AGAIN.
[3.00]
3 F:CORRECT.
[4.00]
- R:
[5.00]
- F: C:
```

To print just Groups 3 and 4 of Frame 84 (without line numbers):

```
*84,3-4
```

```
G 3.  ANSWERS
1 SCP
2 SMD
3+SSD

G 4.  ACTIONS
1 R:YOU ARE STILL READING THE FIRST PROBLEM.  TRY SSD THIS TIME.
2 R:SUM OF THE MEAN DIFFERENCES WAS THE LAST PROBLEM.  TRY AGAIN.
3 F:CORRECT.
- R:
- F: C:
```

To print just line 4 of Group 4 of Frame 84

```
*84,4,4
- R:
```

To print lines 2 and 3 of Group 3 of Frame 84

```
*84,3,2-3
2 SMD
3+SSD
```

To print all lines of all groups of Frame 84

```
*84,1-4,1-5
AND WHAT IS THE SUM OF THE SQUARED DIFFERENCES FOR THIS TABLE?
1 SCP
2 SMD
3+SSD
1 R:YOU ARE STILL READING THE FIRST PROBLEM.  TRY SSD THIS TIME.
2 R:SUM OF THE MEAN DIFFERENCES WAS THE LAST PROBLEM.  TRY AGAIN.
3 F:CORRECT
- R:
- F: C:
```

If the request is to print the entire frame (omission of group numbers and line numbers), PLANIT prints frame header and all group headers. Frame header includes frame type and label, if any. Group headers contain G (for group) followed by the group number and type (concise level). If the request is to print just some groups (omission of line numbers), the group headers are printed out. If the request is for lines only, PLANIT prints out just those lines and their line numbers (no header).

```
*#CO                LD wants to continue building

PROBLEM/QUESTION/MULTIPLE CHOICE/DECISION/COPY

*
```

(#GET) GETTING A LESSON OR STUDENT'S RECORDS

Note throughout the discussion on GET and SAVE, the following applies: The term "alphamerics" means 2-6 letters or numbers, the first of which must be a letter. If any numbers are used, no letters can follow; e.g.,

RIGHT	WRONG
A1235	1ABC
SS4	A
PP	KK52B
PKV32	

Lessons can be saved on either disk or tape. Hence, they can be retrieved from those devices. The general form for getting a lesson is:

GET LLLL DDDD where LLLL is the lesson name (alphameric) and DDDD are integers for the tape number. If the tape number is left off, PLANIT will assume the lesson is on disk. Even though the lesson is on tape, the lesson name must be entered first followed by the tape number if the LD desires to get the lesson from tape. PLANIT will first check to see if it can get the lesson off disk; failing that, PLANIT will load the lesson from tape.

All lessons loaded from tape by the GET command are immediately loaded onto disk as well so that when the LD receives the response "LESSON SUCCESSFULLY LOADED" the lesson will also be on disk.

For example, if you type

GET STAT, PLANIT will look for STAT on disk. If found and loaded successfully, PLANIT will respond with: LESSON SUCCESSFULLY LOADED. At this point the LD can proceed with any of the legal commands. If STAT is not found, the response will be: LESSON NOT FOUND, REPEAT COMMAND WITH REEL NO. The LD might then type: GET STAT 1035 (assuming he has previously saved STAT on tape 1035) and the computer operator will be instructed to mount the appropriate tape reel.

Actually before getting the lesson, PLANIT will ask for identification. At this point you must type the ID (alphameric) under which you are using the lesson. If the ID matches the ID of the designer of the lesson, the program will be in the COMMAND mode upon completion of the load. Otherwise, the lesson will commence to operate or continue from where it previously left off in the execution of the lesson.

PLANIT keeps track of all student's use, so it automatically continues from the last frame the student was on when he "FINISHED" for the day (See CALC 10.2.24.)

The LD (only) can examine student's records. He can do this by saying: GET STAT STD01 where STAT is the lesson name and STD01 is the student's ID. PLANIT will ask for the LD's ID before getting the student's record. Having successfully loaded the record for STD01, the LD then operates as though he is that student except that he retains LD privileges; he can DISPLAY records, examine CALC items, EXECUTE any part, etc. He can also change the record and save the changed record onto disk again with the command, SAVE STAT STD01 (STAT and STD01 are used as examples). In this way he can change the point from which the student will resume on the next session. (See the SAVE command for further details).

(#SAVE) SAVING A LESSON

The SAVE command is similar to the GET command; e.g.,

SAVE STAT PLANIT saves STAT onto disk (after receiving an ID from the LD).

It is the same ID that is looked for when STAT is loaded.

SAVE STAT 1035 PLANIT saves STAT onto tape number 1035.

SAVE STAT01 PLANIT saves STAT as STAT01 on disk.

There are now two versions of STAT on disk.

When the student is finished for the day, he must type in: ←FINISHED (i.e., left arrow followed by FINISHED). If he does not do this, that day's records will not be saved onto disk.

The first time a lesson is built, the LD must save it on disk. Having done so, he can then go ahead and save that same lesson on tape. All subsequent changes to the lesson can then be followed by an immediate SAVE to tape.

All GET and SAVE commands are done through core memory. For example: If the LD has a lesson named AAA in core and wishes to save a lesson whose name is BBB that resides on disk, he must first GET BBB and then follow it with SAVE BBB 1234 where 1234 would be the tape name to which the lesson is assigned. Note in the examples just given, lesson AAA (which was in core at the time of the GET command) was clobbered over by lesson BBB. Naturally, the LD would have first initiated a SAVE AAA command to prevent AAA from being clobbered if he didn't have one already on disk.

Student records can also be saved directly under command of the LD. SAVE AAA STD01 will cause the student record portion of AAA to be saved onto disk for the student with ID of STD01.

(#EX) EXECUTING A LESSON

Having built a few frames, the LD can now execute the lesson to see it as a student would.

The general form of the command is: EX,D,C where EX means EXECUTE, D is a frame number or label, and C means clear all previous records.

NOTE: EX 0,C is a special use of EXECUTE that clears all LD records without beginning execution of the lesson.

The "C" is optional. If used, it causes all previous records to be cleared, causing PLANIT to disregard what happened before. This is important if the LD is checking out some Decision frames. If the "C" is missing, his previous performance can affect the Decision frame branching pattern.*

If the "D" is present, the lesson will commence from Frame D.

EX by itself means that the lesson will continue from the last frame entered. If the user is just starting, he will begin on the first frame.

(#RESTART) ERASE AND RESTART LESSON BUILDING

This command will cause PLANIT to start over. It will clear out all its information tables and return with P/Q/M/D/C.

This command is recognized only when given by the LD.

---

*The presence of C in the EX command only clears the LD records when used with the frame number, e.g., EX 5, C. If the frame number is absent (EX, C), C will be taken to mean a frame label.

(#DISP) DISPLAY STUDENT'S RECORDS

This command can only be used by the LD; indeed, all legal commands can only be used by the LD once a lesson is loaded. (PLANIT will not recognize the # if typed in by anyone other than the LD.)

Let us assume we have some student's records in PLANIT (e.g., as a result of GET STAT STD01). The LD types in:

DISP and PLANIT responds with:

```
******LESSON    STAT *****STUDENT    STD01*****

LABEL     FRAME     TYPE     ANSWER     RIGHT/WRONG     STEPS     TIME

          1.0        Q         A             +            0         1

PROB      2.0        P         0             0            0         0

INTRO     2.5        M         C             -            0         1

INTRO     2.5        M         B             +            0         1

          3.0        Q         -             -            1         5

          3.0        Q         3             +            2         10

**OPTS    SIN   COS   COMB   MEAN

          4.0        Q         A             +            0         3

          5.5        M         A             +            0         2

TIME-MIN 23
```

Most of the data here are self-explanatory; however, a couple of areas need explaining.

If the frame had no label, none is printed out. The time is in minutes (the greatest integer). Time on the bottom is the column total under TIME.

RIGHT/WRONG of a P or D frame is always zero and ANSWER is always zero.

The minus (-) under ANSWER for Frame 3 indicates that the answer given did not match any of the anticipated answers (unanticipated answer).

On the first occurrence of Frame 3.0 the student used one STEPS for the solution. On the second occurrence, he used another step. At this time he also used SIN COS COMB and MEAN functions. Notice that all functions used are listed whether primitives of PLANIT or LD defined.

The student has repeated Frames 2.5 and 3.0 twice in succession.

(#CALC ←) THE CALC MODE

One may enter CALC in either of two ways:

(1) by typing #CALC whereupon CALC will reply OK*, or (2) by prefacing a CALC statement with the symbol,←. Once in CALC, the special symbols need not be used.

Only a lesson designer will have access to the command #CALC which provides automatic protection for what he does in CALC so that the student can neither see nor alter the work unless allowed to do so. Using the ← while in the execution mode puts the user into a restricted CALC mode where he has a subset of the capabilities available to the LD.

The following explanations will apply to both the LD and the student unless otherwise stated.

General definitions

1.0 Names

CALC names include names for temporary constants, set constants, subscripts, functions, dummy arguments, temporary matrices, set matrices, library options and primitive function words. Relatively few names are pre-defined in CALC. Most will be defined by the user.

1.1 Name format

All names must be composed of letters or letters and numbers. Names may contain one or more characters. Any two names whose first eight characters are identical will not be distinguished by CALC. All names must begin with at least one letter. If numbers are used in the name, no letters may follow the numbers in that name.

Correct: I
MEAN1
CORRELATION

Incorrect:	Will be translated as:
12J	(12)(J)
A2B	(A2)(B)
VAR=A	VAR= A

1.2 Name usage

The following names have been concocted to clarify this booklet. These names have no special meaning to CALC. Rather, they will help clarify format specifications.

1.2.1 NEWN A new name nowhere defined in CALC

1.2.2 TMPN A temporary constant name

1.2.3 SETN A set constant name

1.2.4 SUBN A subscript name

1.2.5 FUNN A function name

1.2.6 DUMN A dummy argument name

1.2.7 TMPM A temporary matrix name

1.2.8 SETM A set matrix name

1.2.9 EXPRESSION Any arithmetic expression containing numbers, names or operators in any combination.

These names will be used with reference to the given definitions.

2.0 Temporary and Set distinctions

2.1 Temporary names

Temporary names are those which CALC allocates to a list which can be erased in one sweep.

2.1.1 DROP NAMES erases the above names.

2.1.2 The generation of a new sample in a problem-frame erases the above names.

2.2 Set names

2.2.1 Protected set names are those names which are placed into the SET Classification while operating CALC as a lesson designer (LD) and automatically protected. Students may neither access nor alter these names. The LD can both access and alter them. These names will not be dropped as in 2.1 above.

2.2.2 Unprotected set names are those defined while operating CALC as a student and using the SET option. These names are accessible to both student and LD and will not be dropped as in 2.1 above.

2.3 Creating a temporary name (Also see ASSIGN)

2.3.1 A temporary name can be created by setting it equal to a constant.
NEWN = EXPRESSION

2.3.2 A value can be replaced.
TMPN = EXPRESSION
Incorrect: SETN = EXPRESSION

2.4 Creating a set name

2.4.1 SET NEWN = EXPRESSION

2.4.2 SET SETN = EXPRESSION

2.4.3 SET TMPN

2.5 Matrix names can be set as in:

2.5.1 SET MATRIX (NEWN, N1, N2, . . .)

2.5.2 SET MATRIX (SETM, N1, N2, . . .)

2.5.3 SET TMPN

3.0 Expressions
Expressions are made up of numbers, names or operators or all three.

3.1 Numbers

3.1.1 Size - The maximum number of significant digits will vary with the computer.

3.1.2 Very large or very small numbers can be expanded by powers of ten, e.g., 1234*10**124 and 1234*10**(-204).

3.1.3 A decimal point can appear anywhere in a string of numbers with usual interpretation. No extra leading or trailing zeros are required.

3.1.4 A % character may follow a number causing appropriate conversion to take place. Leave no blank between the last numbers and the %. The % may not follow a name.

3.1.5 Numbers which are printed out are rounded according to usual statistical rules.

3.1.6 Though rounding occurs when the numerical value of a name is printed, the maximum precision is nevertheless retained in the computer for that name.

3.2 Operators

+ add
- subtract or negate
* multiply
/ divide
** exponentiation, e.g., 5^2 becomes 5**2
Adjacent numbers or names of numbers without operators imply multiplication, e.g., (5)(2)
([opening parentheses
)]closing parentheses
= replace contents of name on the left with numerical evaluation of expression on the right
, comma (argument separator)

3.3 Names - as described in 1.0

4.0 Function forms and general format for specifying arguments

4.1 General form
FUNNAME (A1, A2, . . ., AK)

4.1.1 Function name conventions same as other name conventions (see 1.0).

4.1.2 Parentheses are essential.

4.1.3 Commas are essential if two or more arguments are part of the function.

4.1.4 For combinable-type functions (see 4.2), either of the arguments may themselves be expressions, e.g.,
FUNNAME ((A1+2), TMPM(3), (SETN**2))

4.2 Combinable vs. non-combinable functions

4.2.1 Single-valued functions whose arguments are all numbers or names of numbers are combinable functions and can be one of many terms in a longer expression.

These functions may have expressions for arguments provided

the expression names a number, e.g.,
X = SQRT (Y) +2.

4.2.2 Non-combinable functions are those which return more than one value or some message; also those for which one or more arguments consist of names to be interpreted as names rather than the numbers they represent, e.g., HELP (SQRT). Here SQRT is the name supplied as the argument, but SQRT has no value associated with it. It is a function name for the square root function.

4.2.3 Restrictions on the non-combinable functions

4.2.3.1 A non-combinable function, if used, must be the only term of the expression.

4.2.3.2 Non-combinable functions may not be combined with other terms in the same expression. Hence, they must appear alone in a given expression.

right: RANK(TMPM)
wrong: SETM = RANK (TMPM) + K

4.3 The number of arguments is fixed for each function. The argument values must be specified whenever the function is used; e.g., COMB(A,B) is a function yielding the number of B distinct groupings in A things. Both arguments must always appear after COMB.

5.0 Matrices and matrix subscripting

5.1 Predefined matrix names
VALUES N DATA LINK

5.1.1 VALUES will contain a protected copy of the last set of data to be printed in a problem-type frame with appropriate dimensions.

5.1.2 N is a 2x1 protected matrix which contains the group sizes for the data in VALUES.

5.1.3 DATA is an unprotected copy of VALUES (5.1.1).

5.1.4 LINK is a 10x1 protected matrix. This matrix is unique in that its contents are transferred to a new lesson where lessons are chained together.

5.2 All other matrices must be defined by the user using the function

5.2.1 General form
MATRIX (NEWN, A, B, C, D)

Where NEWN will become the temporary name of the matrix, A-D are four dimensions of the matrix (rows, columns, blocks and levels).

5.2.2 Possible omissions
B-D are omitted if all equal one.
C-D are omitted if all equal one.
D is omitted if equal to one.
Parentheses are essential.

e.g., MATRIX(CORREL, 12, 2)

This is a non-combinable function (4.2).
As such, A-D can be numbers, constant names or elements of a matrix.

E.g., if two dimensions are specified for a newly defined matrix, then two subscripts will be expected to follow future use of the name.

5.3 Matrices may be protected from being automatically dropped (see also 2.1 and 2.5) by either of the following statements:

```
    SET TMPM
SET MATRIX (NEWN, N1, N2, . . .)
```

These are non-combinable expressions (4.2).

5.4 Subscripting a matrix
All matrices must be followed by as many subscripts as dimensions specified (5.2.2).
Exceptions:

5.4.1 If one matrix is being filled with values from another matrix with identical dimensions, simply type: A = B where A and B are the names of the respective matrices. This is a non-combinable expression (4.2).

5.4.2 Certain non-combinable functions accept matrix names as arguments without the subscripts since the entire matrix is being referenced, e.g.,

```
HELP (MATNAME)
RANK (MATNAME)
```

5.5 Matrix names can be used as combinable functions in expressions (4.2), but they must always be followed by the correct number of subscripts. Commas provide the means for counting subscripts.

5.6 Size restrictions
Matrices may have from one to four dimensions (5.2.1) and can contain up to 1000 cells.

5.7 Subscripting
Matrix subscripts are integers greater than zero.
The first entry of a four-dimensional matrix is (1, 1, 1, 1).
The same conventions apply for specifying subscripts as for specifying arguments.

5.8 Filling a matrix

5.8.1 The matrices VALUES, DATA and N are filled by the problem frame sample generator.

5.8.2 A matrix may be filled by use of a FOR condition (to be discussed later).

5.8.3 Any matrix cell may be filled with a discrete number, e.g.:
MATNAME (3, Z) = 6.

5.8.4 A matrix may be filled with discrete numbers by typing:
MATNAME (1, 1) = NEWN

(Assuming MATNAME is two-dimensional)
the computer responds:
ENTER VALUE FOR: NEWN
Reply to this by typing in an array of up to 20 numbers (or constant names) separated by commas, e.g.,
6.2, SETN, TMPN, 4.6, etc.
The values will be set into MATNAME in the following way:
The last subscript will increment first until that dimension is full, then the next, etc., as indicated by the subscripts:
MATNAME_{ij} (j=1, ..., n; i=1, ...,m)

A little practice will soon reveal the pattern. The same rule applies to matrices with from one to four dimensions. The subscripts given to the matrix will determine where the filling begins.

5.8.5 A matrix may be filled using the ARRAY function (a non-combinable function, see 4.2).
MATNAME (1,1)=ARRAY(6.2, SETN, TMPN, 4.6, etc...).
Up to 20 numbers may be specified. The subscripts of the matrix determines where the numbers will start filling. The procedure closely resembles 5.8.4.

6.0 User-defined (UD) functions

6.1 All user-defined functions are the combinable type (see 4.2).

6.1.1 As such, all arguments must be either numbers or names of numbers.

6.1.2 All user-defined functions must be single-valued.

6.1.3 The argument format for functions follows the general form (4.1).

6.2 Defining a function

6.2.1 General form:

FUNCTION NEWN (A1, A2, ..., AK) = EXPRESSION

6.2.2 Arguments A1, A2, ..., AK are dummy names which apply only to their occurrence in the EXPRESSION to the right, but not to the context of the EXPRESSION that calls this function.

6.2.3 Arguments A1, A2, ..., AK must each represent a name for a single number.

6.2.4 From Zero to 20 arguments may be specified for any one function.

6.2.5 No equal sign (beside that appearing in 6.2.1) may appear in the EXPRESSION unless it follows a FOR. (i.e., functions definitions may not be used to define or set values into items except as indicated in FOR statements.)

6.2.6 No non-combinable functions may appear in a user-defined function (see 4.2).

6.2.7 Combinable-type functions may appear in the EXPRESSION part of the function definition.

2

6.2.8 No function may call on itself (i.e., it cannot be defined recursively in terms of itself).

6.2.9 Example

Right:

```
  FUNCTION POWER (X,Y) = X**Y
  FUNCTION PERM (C1, C2) = FACT (C2)*COMB (C1, C2)
```

Wrong:

```
  FUNCTION SETX = X = FRAME*100
  FUNCTION LIST = PRINT NAMES
  FUNCTION SETUP = RANK (MATRX)
  FUNCTION RECUR(X) = X*RECUR(X-1)
```

6.3 Using a UD function once it is defined

6.3.1 The UD function is called by its name plus the exact number of arguments of its definition.

6.3.2 Arguments in a UD function call must each be numbers or names of numbers.

6.3.3 Arguments in a UD function call may be numbers, constant names, elements of a matrix, other functions or EXPRESSIONs containing these.

6.3.4 Arguments in a UD function call may not be an unsubscripted matrix name.

6.3.5 Examples

Right:

```
  POWER (9,3)
  POWER ((MATNAME(3), (COL-2))
```

Wrong:

```
  POWER (MATNAME,2)
```

6.4 All user-defined functions are kept in the SET area and will not be automatically dropped (see 2.1 and 2.2).

6.5 All functions defined by an LD are automatically stored in the protected set area (2.2.1). See introduction for entering CALC as a LD.

6.6 User-defined functions cannot exceed one line.

6.7 Except for the above restrictions, the EXPRESSION in a user-defined function may be any EXPRESSION that CALC will allow outside of a function.

7.0 Subscripting

7.1 A subscript may be any legally constructed name.

7.2 If a TMPN is used as a subscript, it will retain its status as a temporary constant after evaluation of the EXPRESSION is complete. Its value will be the initial value it assumed as a subscript.

7.3 Subscripts are constant names that assume different values as the EXPRESSION is evaluated. They may be used in the same way other constant names are used.

7.4 Standard name conventions also apply to subscripts.

7.5 Subscripts are defined by a FOR term and remain defined as subscripts only during the evaluation of the EXPRESSION in which they are defined.

8.0 FOR terms

8.1 General form for defining subscripts:

```
FOR (J1=A1,A2 J2=B1,B2 J3=C1,C2 etc.)
```

8.1.1 J1, J2 and J3 can be any name.

8.1.2 A1, A2, B1, B2, C1, C2 may be any integer value or name of an integer. If the number is not an integer, it will be truncated to the first integer value less than the original number (i.e, the greater integer).

8.1.3 From one to many subscripts may thus be defined. The general form shows three being defined.

8.1.3.1. Examples of one being defined:

```
FOR (J1=A1, A2)
FOR (I=1, 10)
```

8.2 Interpretation of the arguments

8.2.1 Using the example: FOR (J1=A1, A2)
J1 is the subscript name.
A1 (which may be an expression) is the initial value of J1.
A2 (which also may be an expression) is the last value of J1.

8.2.2 Increment (use the example in 8.2.1)

8.2.2.1 If A1<A2, J1 will start at J1=A1 and increment by one until J1 equals A2.

8.2.2.2 If A1>A2, J1 will start at J1=A1 and decrement by one until J1 equals A2.

8.2.3 If only one argument, A1, is given, J1 will be set to A1 and will not range. The statement will be evaluated once for J1=A1.

8.2.4 If three arguments, A1, A2, and A3, are supplied, J1 will equal A1, then A2, then A2+(A2-A1), ..., A3.

8.3 EXPRESSION field over which the subscript ranges

8.3.1 If no SUM or PROD operators appear in the expression, the entire expression to the left of the definition of the subscript name will be included in the field of operation.

Usually when no SUM or PROD operators are used, the FOR (SUBN=1,10) term appears at the extreme right of the expression.

8.3.2 Typical uses of the FOR(   ) term when no SUM or PROD operators appear.

8.3.2.1 Filling a matrix
The following example shows a two-dimensional matrix being filled with the product of the subscript values.

SETM(I,J)=I*J FOR (I=1,4 J=1,4)

After this EXPRESSION has executed, the resulting matrix SETM would contain these values:

$$\text{SETM}= \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 6 & 8 \\ 3 & 6 & 9 & 12 \\ 4 & 8 & 12 & 16 \end{bmatrix}$$

8.3.2.2 Transferring the contents of a matrix to another matrix. Suppose TMPM is a 4x1 matrix and is to be filled with the diagonal elements from SETM (in 8.3.2.1). This EXPRESSION would accomplish that:

```
TMPM(I) = SETM(I,I)  FOR(I=1,4)
```

After executing:

$$\text{TMPM}= \begin{bmatrix} 1 \\ 4 \\ 9 \\ 16 \end{bmatrix}$$

Using similar EXPRESSIONS, matrices may conveniently be manipulated.

The cells of a matrix may be filled with an exact copy of an identically declared matrix as is shown for two sample matrices:

```
MATNAME= SETM
```

Where the dimensions of the two matrices are exactly alike.

8.3.3 If a SUM or PROD operator appears in the EXPRESSION, the subscript is defined over that part of the EXPRESSION between the SUM or PROD operator and its associated subscript initialization. A SUM or PROD operator must always have a FOR term to its right in which a subscript is defined for each SUM or PROD operator.

If more than one subscript is defined over the same part of the EXPRESSION, the subscripts increment from innermost to outermost.

9.0 SUM and PROD operators

9.1 The SUM operator represents as nearly as possible the Σ in mathemmatical notation. Since the range of summation cannot be represented as mathematical superscripts and subscripts, the FOR term is used instead. For example:

$$\sum_{i=1}^{N} (X_i - Y_i)$$

becomes:

```
SUM(X(I)-Y(K)) FOR (I=1, N)
```

But since the range of the subscript is defined between SUM and FOR, it is equally permissible to write:

```
SUM X(I)-Y(I) FOR (I=1,N)
```

9.1.1 Parentheses must be consistent.
The part of the EXPRESSION between the SUM and the FOR is implicitly a parenthesized group. Therefore, both members of each pair of parentheses must be either inside or outside of this area.

Right:

```
  SUM (X(I) - Y(I))  FOR (I=1,N)
  (SUM X(I) - Y(I) FOR (I=1,N))
```

Wrong:

```
  (SUM X(I) - Y(I)) FOR (I=1,N)
  SUM (X(I) - Y(I) FOR (I=1,N))
```

9.1.2 One or more SUM or PROD operators may appear in an EXPRESSION. Each such operator is paired with the next successive subscript initialization to the right and the operator is valid over that part of the EXPRESSION that lies between the two.

9.2 PROD

The PROD operator represents the π mathematical symbol used for accumulating products. The conventions for the use of PROD are identical to SUM (see 9.1).

10.0 Preset names (primitives of the PLANIT program)

10.1 Names listed in response to PRINT NAMES

10.1.1 N a 2x1 protected matrix (see 2.2.1)
N(1) equals group size of column one of the sample data.
N(2) equals group size of column two of the sample data.
These values are updated whenever a problem frame generates a new sample or when the RESET VALUES function is used (see 10.2.21).

10.1.2 VALUES matrix (protected, see 2.2.1)
The VALUES maxtrix holds a copy of the data that is generated by a problem frame.
VALUES is a N(1) x 1 matrix when N(2) is zero.
If N(2) is greater than zero, VALUES has max(N(1), N(2)) rows and two columns.
If N(1) ≠ N(2), the unused matrix elements are zeros.

The VALUES matrix is automatically redefined every time new sample data is generated from the parameters in a Problem frame.

The dimensions of VALUES may also be changed by using the RESET VALUES option (see 10.2.21).

10.1.3 LINK matrix (protected, see 2.2.1)
The LINK matrix is a 10x1 protected matrix, the dimensions of which are fixed. This matrix is unique in that when a lesson temporarily terminates and another lesson is loaded as a follow-on lesson, the ten values in the LINK matrix are transferred to the LINK matrix of the new lesson to be used at the LD's discretion. These are the only values thus transferred.

10.1.4 DATA matrix (unprotected, see 2.2.2)
Whenever data is generated in a problem frame filling the VALUES matrix, an unprotected duplicate of that matrix is also created and named DATA. (See 10.1.2 for its dimensions which are identical to VALUES.)

The dimensions of the DATA matrix may be redefined by using the RESET DATA option (see 10.2.21).

DATA can at any time be restored to an exact duplicate of the VALUES matrix by using the RESTORE option (see 10.2.1).

10.1.5 The following names comprise the library of primitive functions that may be controlled by the LD in Group 3 of a problem frame (in the same way that he controls those of his own making).

Note: The following functions that operate on the sample data use the entries in the matrix DATA if the option is being used in the student mode. If used while executing frames, the entries of VALUES will be used.

10.1.5.1 STEPS (non-combinable, 4.2)
Causes the printout of the next step to the solution as specified in Group 4 of the problem-type frame.

10.1.5.2 HELP (ANYNAME) (non-combinable, 4.2)
Any one of the CALC names may appear after HELP yielding some message explaining the status of that name.

10.1.5.3 RANK (MATNAME) (non-combinable, 4.2)
RANK replaces the entries of any matrix designated by MATNAME with their corresponding rank values. Tied ranks will receive averaged rank scores.

10.1.5.4 SQRT (EXPRESSION) takes the positive square root of that expression.

10.1.5.5 COMB (M,N) is equivalent to the mathematical function:

$$\binom{M}{N} = \frac{M!}{N!(M-N)!}$$

10.1.5.6 FACT (M) is equivalent to M factorial.

10.1.5.7 LOG (M,N) is the logarithm of M taken to the base N i.e., $\log_N (M)$

10.1.5.8 LN(M) is $\log_e(M)$, the log of M to the base e.

10.1.5.9 The six trig functions: SIN(X), COS(X), TAN(X), COT(X), SEC(X), CSC(X) have their usual mathematical connotation.

The argument X will be interpreted as degrees unless RADIANS has been used in that or a previous EXPRESSION (see 10.2.2 and 10.2.3).

10.1.5.10 ABSOLUTE (EXPRESSION) is an absolute value function in the usual mathematical sense.

10.1.5.11 TRUNCATE (EXPRESSION) is a function which rounds the EXPRESSION downward to the nearest integer, dropping the fractional part.
Do not break words at the end of lines.

10.1.5.12 The following four options are table lookup functions for retrieving values from standard statistical tables.

ZTOP(ZCORE) - Normal cumulative distribution function.
PTOZ(PROBABILITY) - Inverse of ZTOP.
PTOT(ALPHA, DEGREES OF FREEDOM) - Student's T-table
PTOX(ALPHA, DEGREES OF FREEDOM) - Chi-square table.

10.1.5.13 INVERT(MATRIXNAME) (non-combinable, 4.2) This function does a matrix inverse operation on the N x N matrix denoted by MATRIXNAME. If the matrix is singular or not of dimensions N x N, an error will be reported. The inverse of the designated matrix takes the place of the original matrix; the name is unchanged but the original values will have been destroyed, being replaced by the inverse values.

10.1.6 Any new names defined by either a LD or student will be added to this list.

10.2 Names listed in response to PRINT OPTIONS

10.2.0 Some of the options listed below are undefined if used as a part of a longer expression. Most of these will be obvious by their nature. These options will be identified by the word "non-combinable," (see 4.2). The following are unprotected:

10.2.1 RESTORE (non-combinable, 4.2)
Changes the DATA matrix into an exact duplicate of the VALUES matrix (see 10.1).

10.2.2 DEGREES changes the CALC format so it interprets all trigonometric function arguments as degrees. This applies for any such arguments to the left of DEGREES in that EXPRESSION as well as succeeding EXPRESSIONS, e.g., SIN (180) DEGREES.

10.2.3 RADIANS does exactly as 10.2.2 except that arguments are then interpreted as radians, e.g., SIN (6.28) RADIANS. (Note: RADIANS and DEGREES need only appear when one desires to change from one to the other.) Trigonometric functions are initially set for degrees.

10.2.4 CHANGE ANYNAME TO NEWN (non-combinable, 4.2)
Any name listed in 10.0 may be given a synonomous new name. The old names will still be valid and can be used except when TMPN names are changed (TMPN will not be saved). The NEWN thus assigned will have all of the attributes of the old name.

10.2.5 SET TMPNAME
(non-combinable, 4.2)
Any previously defined temporary name can be transferred to the SET area using this option (see 2.0).

The name may be set at the same time as it is defined and is the only means of changing a value of a constant name, e.g.,

```
SET SETN=EXPRESSION
```

10.2.6 TO (see 10.2.4)

10.2.7 PRINT (non-combinable, 4.2)
PRINT produces a listing on the teletype whether used in CALC or the execution mode. If used in the student CALC mode, only the unprotected and ALLOWED names will appear. If used in the LD CALC or execution modes, all names will appear.

PRINT NAMES produces the names under 10.1.

PRINT OPTIONS produces the names under 10.2 and 10.3.

PRINT MATNAME produces a listing of the contents of the entire matrix.

PRINT CONSTANT produces the value of that constant.

10.2.8 DROP ANYNAME (non-combinable, 10.2.0)
Drops the name and its synonyms.
Original primitive names will not be dropped.
LD defined names cannot be dropped by the student.

DROP NAMES drops all of the temporary constants and temporary matrices (see 2.0).

10.2.9 PLACES
Specifies the format of the maximum number of fractional decimal digits that will appear on the teletype from CALC.

The number used to set PLACES may be a CALC expression, e.g.,

```
N+2 PLACES
```

The setting of PLACES has no effect on the precision of the number retained in the computer. It only determines the form in which the number will be printed.

Consider this example:

```
*4 PLACES
IN

*N=1/3
N=0.3333

*6 PLACES
IN

*N
0.333333
```

The final digit will be rounded according to usual statistical practices.
(It is rounded up only if the preceding digit is odd-numbered.)

Only the first zero of a string of non-significant zeros will appear, e.g.,

```
*6 PLACES
IN

*1/2
0.50
```

Zero PLACES implies rounding to the nearest integer.

```
*1 PLACES
IN

*N=3/2
N=1.5

*0 PLACES
IN

*N
2
```

10.2.10 NAMES (see PRINT, 10.2.7, DROP, 10.2.8 and ASSIGN, 10.2.11) (non-combinable, 4.2)

10.2.11 ASSIGN NAMES (non-combinable, 4.2)
After using this option, CALC will define a name and fill it with the numerical evaluation of any expression that is not otherwise set into a constant name.

To disable this option, type DONT ASSIGN NAMES.

10.2.12 FOR (see 8.0 and 9.0)

10.2.13 MATRIX (MATNAME, DIM1, DIM2, DIM3, DIM4) (see 5.2) (non-combinable, see 4.2)

10.2.14 ARRAY (A1, A2, A3, ..., A20)
(non-combinable, see 4.2)
Used for filling a matrix. (see 5.8.5)

10.2.15 READY (non-combinable)
Use for leaving CALC. Returns to the mode from which CALC was entered.

10.2.16 OPTIONS (see 10.2.7)

10.2.17 VERBOSE
Will change the amount of information provided by PLANIT by using VERBOSE in the EXPRESSION, e.g.,

BE VERBOSE

PLANIT will then only give long messages in the lesson building mode.

10.2.18 CONCISE
Same conventions as VERBOSE (10.2.17) except that a short message format will result.

10.2.19 SUM (see 9.1)

10.2.20 PROD (see 9.2)

10.2.21 RESET (non-combinable, 4.2)
General form:
RESET (VALUES, N1, N2) or
RESET (DATA, N1, N2)

This function allows one to change the dimensions of the two named matrices. N1 refers to the number of elements of data in the first data column, N2 refers to the number of elements, if any, in the second data column. If N2 is zero, it may be omitted.

The new matrix will have as many rows as the larger of N1 or N2 and one or two columns, depending on whether N2 is greater than zero.

Since LIBRARY options may use the matrices DATA and VALUES, the defined group sizes may be important.

Notice that N1 does not have to equal N2.

10.2.22 FUNCTION (see 6.0)

10.2.23 RANDOM
RANDOM is a primitive name that can be used in the same way as a constant name. RANDOM will randomly assume values from 0.0 to 0.99999 uniformly distributed.

10.2.24 FINISHED (non-combinable, 4.2)
FINISHED terminates the student and saves his records.* When he restarts and GETs the lesson again, he will resume on the frame where he left off.

10.3 This group of primitive names are protected LD options and cannot be used by the student unless the LD has permitted its use by the ALLOW option (see 10.3.3). (These names are also listed in response to PRINT OPTIONS).

10.3.1 RELATED (non-combinable, 4.2)
The expression RELATED ON turns on the option of selecting group 4 actions on the basis of related responses that were given before in the same frame. See the lesson building section for further details.
RELATED OFF turns off this capability.

10.3.2 STATUS (non-combinable, 10.2.0)
STATUS allows the LD to query the status of certain control features of the lesson segment.
STATUS may be followed by any one of the following names:

---

*After typing in FINISHED there will be a pause of one or two minutes. During this time PLANIT will update the student's records and put them on disk. Upon completion PLANIT will respond with PROGRAM CONCLUDED.

KEYWORD, PHONETIC, FORMULAS, RELATED, WITHIN, WAIT, or PLACES.
If only the word, STATUS, is typed, all of the above conditions will be reported and, in addition, a figure will be given representing the percentage of the lesson segment space that has been filled.
E.g.:

```
*STATUS
KEYWORD OFF, PHONETIC OFF, FORMULAS OFF, RELATED OFF,
WITHIN OFF, WAIT OFF, 4 PLACES, SPACE 63% FULL
*
```

10.3.3 REEL (see 10.3.7)

10.3.4 ALLOW ANYNAME (non-combinable, 4.2)
This option allows the student to use any protected name (see 2.2.1), although he can neither CHANGE nor DROP it.

10.3.5 KEYWORD (non-combinable, 4.2)
The expression KEYWORD ON turns on the capability of examining answers for keywords. See the lesson building section for further details.

KEYWORD OFF turns off this capability.

10.3.6 PHONETIC (non-combinable, 4.2)
The expression PHONETIC ON turns on the capability of accepting misspelled answers. See the lesson building section for further details. PHONETIC OFF turns off this capability.

10.3.7 GOTO (LABEL)
In general one can use GOTO in the form

```
GOTO XXXX      or      GOTO XXXX REEL(1234)
```

In the absence of the REEL parameter, the statement will be interpreted in the following order:

1. If XXXX is a number, control will pass to the frame bearing that number or, if none, an error message will result.

2. If XXXX is a CALC name for a number, that number will be interpreted as (1) above.

3. If XXXX is not a CALC name and a frame label exists by that name, control will transfer to that name.

4. If XXXX is none of the above but exists on disk as a lesson name, control will transfer to the first frame of the new lesson.

5. If XXXX exists as a binary program on disk, control will transfer to that program and back to CALC at the conclusion of that program.

6. If XXXX is none of the above, an error message will result. If the REEL parameter follows the GOTO (LABEL), then the LABEL will be taken to mean a lesson segment name and the reel number will be put in the place of '1234.'

10.3.8 FORMULAS (non-combinable, 4.2)
The expression FORMULAS ON turns on the capability of identifying algebraically equivalent answers. See the lesson building section for further details. FORMULAS OFF turns off this capability.

10.3.9 WITHIN sets the tolerances for anticipated numerical answers in building a lesson. (see example on p. 21, Frame 6.)

10.3.10 PROHIBIT
PROHIBIT restores the original status of a protected word that the LD has ALLOWED (see 10.3.2).

10.3.11 TIME
TIME is a name that holdsthe number of minutes that have elapsed since midnight. It is accurate to the nearest tenth of a minute.
TIME can be used in the same way as a constant name. Latencies are easily available using TIME. At the beginning of the period to be measured, e.g.

SET SAVETIME = TIME
At the desired measuring point,
SET LATENCY = TIME -SAVETIME

10.3.12 FRAME is used as a constant name and holds the frame number of the lesson frame at which the student is using CALC. FRAME is useful for identifying rows of special records being kept in matrices, etc.

10.3.13 WAIT presets a maximum waiting period for the next group 3 student's answer. If the waiting period is exceeded, the lesson will proceed, identifying the lack of an answer with a prime (') tag (see the lesson building section for details).

WAIT is followed by a time parameter which is assumed to be seconds unless followed by the word, MINUTES. For example:

WAIT (15) means wait 15 seconds for the answer to the next question.

WAIT (2 MINUTES) specifies a 2 minute waiting period for the answer to the next question.

10.3.14 RESPONSE is a constant name that contains the value of the last numeric answer that the student gave to a lesson question.
RESPONSE is useful if the expected answer is a number in which case RESPONSE will be the number that the student entered.

10.4 The following names are used in the lesson building mode but they exist in CALC so the LD may CHANGE any of them. (See 10.2.4)

F, C, R, B, P, D, I, CALC, CONTINUE, EXECUTE, GET, SAVE, DISPLAY, RESTART, FROM, NONE, ADD, ALL, BUT, ONLY, USED, NOT RIGHT, WRONG, SEEN, MINUTES.

11.0 CALC EXPRESSIONS more than one line long
If your EXPRESSION requires more than one line, plan your EXPRESSION so that each line breaks after one of the following arithmetic symbols:

plus +, minus -, asterisk *, slash /, or open parentheses (.
The symbols will be considered in the evaluation of the composite EXPRESSION.

12.0 Examples (Possible dialogue with CALC).

1. Compute an arithmetic statement

```
*2 + 2

4.0
```

2. Use function names

```
*SQRT (19 + 6) + FACT (3) + COMB (4,3)

15.0
```

3. Assign names

```
*X = 10 + SIN (45)

X = 10.707

*Y = X - 5

Y = 5.707
```

4. Change names

```
*CHANGE FACT TO FACTORIAL

DONE

*FACTORIAL (4)

24.0
```

5. Cause the computer to assign names

```
*ASSIGN NAMES

WILL DO

*FACTORIAL (4)

LI = 24.0
```

6. Drop names

```
*DROP X

GONE
```

7. Ask for values not given

```
*Y + Z1 + Z2

ENTER VALUES (COMMAS BETWEEN) FOR:  Z1, Z2

#3, 4

L2 = 12.707
```

8. Change the precision

```
*2 PLACES

IN

*L2

L3 = 12.71
```

9. Print all options which can be concurrently used

```
*PRINT OPTIONS
RESTORE  DEGREES  RADIANS  CHANGE  SET  TO  PRINT  DROP  PLACES
NAMES  ASSIGN  FOR  MATRIX  ARRAY  READY  OPTIONS  VERBOSE  CONCISE
SUM  PROD  RESET  FUNCTION  RANDOM  FINISHED  RELATED  STATUS  REEL
ALLOW  KEYWORD  PHONETIC  GOTO  FORMULAS  WITHIN  PROHIBIT  TIME
FRAME  WAIT  RESPONSE  F   C   R   B   P   D   I   CALC  CONTINUE
EXECUTE  GET  SAVE  DISPLAY  RESTART  FROM  NONE  ADD  ALL  BUT
ONLY  USED  NOT  RIGHT  WRONG  SEEN  MINUTES
```

10. Print all names which can currently be used

```
*PRINT NAMES
N  VALUES  LINK  STEPS  HELP  RANK  SQRT  COMB  FACT  LOG  LN  SIN
COS  TAN  COT  SEC  CSC  ABSOLUTE  TRUNCATE  ZTOP  PTOZ  PTOT  PTOX
INVERT  DATA
```

11. Define a matrix (maximum four dimensions)

```
*MATRIX (ABC, 10, 10, 10)

IN

*ABC (1,1,1) = L2

ABC= 12.71

Z = ABC (1,1,1) + Y

Z = 18.42
```

12. Set values or matrices into a protected area

```
*SET Z

DONE

*SET ABC

DONE
```

13. Drop all temporary names

```
*DROP NAMES

GONE
```

Now Y, L1, L2, L3 have all been erased. Only Z and matrix ABC remain together with the primitive names.

14. Store and use functions. (Any number of arguments are legal but the function must be single-valued.)

```
*FUNCTION XYZ (X) = 2X + 4

IN

*Y = XYZ (3) + Z + .58

Y = 29.0
```

15. Do summation, products and transformations

(The word DATA will be predefined by the program and the current values from the most recent problem will be preset.)

The following would produce a vector of standard scores.

```
*MEAN = SUM  DATA (I, 1) / 10  FOR(I = 1, 10)

MEAN = 30.50  (if the values were right)

*SUM  DATA (I, 1)  **2 / 10  FOR (I = 1, 10)

L1 = 1330.25

*STDDEV = (L1 - MEAN**2) ** (1/2)

STDDEV = 20.0

*MATRIX (ZSCORE, 10, Z)

IN

*ZSCORE (I) =(DATA (I, 1) - MEAN) / STDDEV FOR (I = 1, 10)
```

16. Set values permanently for the duration of the lesson (or until they are specifically dropped).

```
*SET PI = 3.14159

PI = 3.14

*6 PLACES

IN

*PI

L2 = 3.141590
```

17. Stop the automatic name assignment

```
*DONT ASSIGN NAMES

NO MORE NAMES

*PI

3.141590
```

18. Change the meaning of the trig. function arguments

```
*SIN (PI / 4) ** 2 RADIANS + COS (45) ** 2 DEGREES

1.0
```

19. He may change the name of any of his command list.

```
*#CALC

OK* CHANGE CALC TO COMPUTE

DONE

*# COMPUTE

OK*
```

20. When he permanently stores names of variables, matrices or functions, they are put in an area which is not accessible to the student. The LD can use these names in providing anticipated numerical answers to questions.

```
*FUNCTION MEANDIFF = SUM (DATA (I,1) - DATA (I,2)) / N(1) FOR
(I=1,N(1))

IN

*MEANDIFF

-2.0

The student mode cannot access that name

*ALLOW MEANDIFF

WILL DO
```

Now the student has access to the function, MEANDIFF.
In this way the LD can build new options for student use if he so desires.

The LD may leave this mode by typing another command, e.g., #CONTINUE.

Many possibilities are probably evident to the reader such as mixing functions, matrices and numbers in a single expression.
Your own ingenuity can fill in from here.

CONDENSED LESSON BUILDING INFORMATION: A SUMMARY

LEGAL COMMANDS (#)

The first of any group of legal commands must be prefaced with a # character. Succeeding legal commands may be written without the #.

Legal commands perform four functions: switch operating modes, manipulate a lesson, edit the text of a lesson, and display student records.

- Switching mode of operation

#CO #CONTINUE	places PLANIT in the lesson-building mode at the next available sequential frame number. (#CO is used to start building a lesson also.)
#EX #EXECUTE	places PLANIT in the execution mode so that the existing lesson will operate as the student will see it.
#RESTART	clears out data tables and starts over. PLANIT responds with P/Q/M/D/C.
#CALC	places PLANIT in the computational mode and makes available an unrestricted use of CALC. See the description of CALC for added detail. Note: #CALC and ← both cause one to enter the computation mode but ← is the more restricted student form.

- Manipulating an existing lesson

#GET MATH #SAVE MATH	moves a lesson from disk into PLANIT or from PLANIT to disk. The name of a new lesson is assigned by the lesson designer (e.g., MATH). The lesson name may have two to six characters, must start with a letter, and may terminate with numbers if desired. Note: Duplicate copies of a lesson can be saved simply by assigning a unique name to each.
#GET MATH 1234 #SAVE MATH 1234	moves a lesson from magnetic tape into PLANIT or from PLANIT to magnetic tape; otherwise, the same as above. The desired tape reel number is specified in place of 1234.
#GET MATH SMITH	Transfer the lesson MATH as used by student SMITH into PLANIT so that the records may be displayed. The LD must give the correct identification for both himself and the student.
#SAVE MATH SMITH	saves student Smith's record of MATH lesson onto disk.
#SAVE	saves a previously named lesson back onto disk.

In performing GET and SAVE operations the user will be asked to identify himself. The same rules hold for the identification of a lesson name: two to six or less characters, all letters, or letters followed by numbers.

- Editing the text of a lesson
  Lines may be inserted into a lesson, deleted from a lesson, or just printed on the teletype by specifying the line or lines and following that with an I,D, or P, respectively (Insert, Delete, Print). The critical part is specifying the line or lines. In the following examples a P will denote print but remember that an I or D could be used just as well for appropriate actions to occur.

  - Specifying a single line: #1, 2, 4, P (i.e., print line 4 in Group 2 from Frame 1).
  - Specifying several lines: #1, 2, 1-5, P (i.e., print lines 1 through 5 in Group 2 from Frame 1).
  - Specifying entire groups: #1, 2, P (i.e., print Group 2 from Frame 1).
  - Specifying several groups: #1, 2-4, P (i.e., print groups 2 through 4 from Frame 1).
  - Specifying an entire frame: #1, P (i.e., print Frame 1).
  - Specifying several frames: #1-4, P (i.e., print frames from 1 through 4).
  - The specific examples above have a general form which is:
    #FRM1 - FRM2, GRP1 - GRP2, LINE1 - LINE2, COMMAND (I, D, P)

    where those frames, groups, and lines included in the specification are affected. Note the effect (above) of omitting certain optional parts of this general form.

    Note: The absence of a command letter will be interpreted as a print (P). In the absence of a P, lines will be printed without line numbers.
    Frame labels may be substituted for frame numbers where desired.

    For inserting, specific frames, groups or lines must be designated. Only one entity can be specified in an insertion command: one frame, one group or one line.

- Displaying student records
  #DISP } prints the records accumulated on the student who was
  #DISPLAY } identified in the GET MATH SMITH command above.

  #DISP FRM1 - FRM2 } same as above but for a subset of the records
  #DISPLAY FRM1 - FRM2 } that occur between the specified frames.

THE FRAME TYPES P/Q/M/D/C

The following paragraphs will explain what information to supply in filling in groups for the five types of frames, but remember, you are not required to supply an entry for every group. If you wish to skip any group, type a space and return. If you have finished building a frame before the groups are exhausted, type #CO or a dollar sign ($).

- (P) The Problem Frame. Nine groups (frame header always Group 1).
  Group 2. Currently not in use

  Group 3. Option control ALL/ALL BUT (TAN,1-3),etc.

  Group 4. Currently not in use.

  Group 5. Steps to the solution. Each comment separated by a semicolon constitute separate steps. These will be printed by student request, one comment for each request.

  Group 6. Sample structure (O)ne, (T)wo independent groups, (M)atched groups, also student preference Y/N. The "Y" may be followed with a message requesting the sample size.

  Group 7. Sample distribution (information requested depends on the reply to Group 6.)

  Group 8. Sample parameters (information requested depends on reply to Groups 6 and 7).

  Group 9. Sample header. The first line is for the data header remarks. The second line is for column location (place any character where the right edge of the column should line up). Remember that the first column will always be a tally column, numbering the rows of the sample. To delete this column, type a NO in place of its location. For example:

```
*     BOYS -- GIRLS

* NO    X       X
```

If fractional data entries are desired, use the format, X.XX, where the number of X's past the decimal determine the precision of the number.

. (Q) The Question Frame

Two types - Constructed Response, POS/NEG

Constructed Response

Four groups

Group 2. Any information or questions can be put in here.

Group 3. All anticipated answers (in the following form):

L ANS1 L is an answer tag and can be any letter.
N ANS2 N is an answer tag and can be any number.

The answer tags must be A-Z or 0-9. Identical tags may be assigned if the answers so designated are not to be distinquished from each other. A plus (+) sign immediately adjacent to any answer tag indicates that a student response which matches that answer will be counted correct. Several plus signs may thus be used.

Answer tags designate three functions:
Tags A-Z imply that the answer will be matched exactly (or according to PHONETIC and KEYWORD rules if these functions are used. See below).

Tags 1-9 imply that the designated answer is a function of the arithmetic evaluation of the line. Simply stated, the line is put through CALC first. For any statement producing a result, that result will be compared to the student's response. Any legal CALC statement may thus be implemented. KEYWORD ON, KEYWORD OFF, PHONETIC ON, PHONETIC OFF are four such legal CALC statements which though they produce no answer to be matched, set up methods for subsequent answer matching. Any CALC line to be matched may have the desired error tolerance specified by WITHIN (e.g. by WITHIN .01 where the number represents the precision demanded in the answer).
For example:

```
1+  MEAN (1)  WITHIN  .001
```

The answer to be matched is the numerical result of the MEAN function and must be exact within the allowable specified error to be counted correct.

Other options which affect the processing of a student response include FORMULAS and WAIT. See pages 21 and 26 for further details.

Most CALC statements will not cause messages to appear on the teletype as they normally do in CALC. However, exceptions to this are the CALC options PRINT and HELP, which will cause the corresponding messages to appear.

Tag 0 should be interpreted in the same way as Tags 1-9 except for two important differences: (1) the associated CALC statement will be done before the student responds to that frame rather than after; and (2) the result of the CALC expression will not be matched to the student's reply.

All answer tags (except for Tag 0 are executed in order from top to bottom, regardless of any other sequence ordering, until a match, if any, is found. No further answers will be executed in that group after a match is found.

Group 4. Action set. There are four basic commands F:, R:, B:, C:. Each of the four commands can be mixed as desired on a given line of actions. Each can also have entries following the colon (e.g., F: YOU AREN'T EVEN CLOSE. B: 5).

In describing how these commands operate with and without entries following the colons, remember that the next command is not considered to be in the expression that follows a previous colon. In the above example, B is not included in the message, YOU AREN'T EVEN CLOSE. The colon following the B guarantees that B will not be considered in the previous message.

F: For specifying feedback. If no message follows F:, PLANIT chooses one to present to the student.

R: Provides for specifying feedback messages as does F:, but then waits for another answer. If no message follows R:, PLANIT tells the student, "WRONG TRY AGAIN."

B: Branch to frame number, frame label, lesson name, program name, or a CALC name of a frame number. Lesson names can be followed by a tape reel number, e.g.

B: LSNAME REEL(1234)

B: with nothing following it, is legal only when used as a "return" to previous lesson.

C: With nothing following it, causes PLANIT to print "THE CORRECT ANSWER IS: (the correct answer)." The only thing that can follow C: is a CALC statement. This allows the LD to include CALC statements in the lesson (apart from the anticipated answer set, group 3).

If these commands are preceded by the letter/number tag associated with the answer, they are performed only if the student's reply matched an answer bearing that tag. Commands preceded by a minus sign are executed only if no match was found, i.e., for unanticipated answers. Commands not preceded by any character (i.e. leading off in the group) are done in all cases. Actions for a given tag (e.g., F: can run over onto one or more lines if each line begins with an action command.

Tags occurring more than once means the action associated with the first occurrence is done first time through frame. Second occurrence is done second time through frame, etc.

The RELATED option, if ON, causes a different interpretation of repeat occurrances of answer tags; namely, that the commands occurring after the second occurrance will be done only if the tagged answer was given a second time in response to that frame.

. POS/NEG form of a Question frame. Has five groups.

Group 2. Same as the conventional Question frame.

Group 3. General form: AAAA/BBBB where A's and B's indicate only two possible single-word answers. Unless specifically POS/NEG then any positive or negative answer is acceptable.

Group 4. LD must specify conditions under which AAAA is true. If conditions not met, BBBB is true.

The conditions for specifying truth is done by IF statements using CALC forms, e.g.:

```
IF 1+1 EQ 4 OR CONT LQ FACT(33) AND...etc.
```

Group 5. Action frame same as group 4 of the conventional Question Frame except that "+" and "-" are arbitrarily assigned answer tags (correct and incorrect). E.g.,

```
+F:  B: AHEAD (to be done if student is correct)

-R:  RECALCULATE (to be done if student is wrong)
```

- (M) The Multiple Choice Frame

  The Multiple Choice frame is built in exactly the same way in nearly all respects as the Question frame. There are only two important distinctions: (1) there are no unanticipated responses. PLANIT will require the student to "CHOOSE ONE OF THE ABOVE LETTERS." (2) Only the answers bearing lettered answer tags will be printed as choices. Numbered answers will be executed so that work in CALC may be done but these will not be matched with student replies. PHONETIC and KEYWORD functions do not apply for multiple-choice frames. The + sign is used in the same way as in the Question frame to denote the correct answer(s) but will not be printed with the alternative answer-choice with which it is associated. Periods may follow answer tags for sake of appearance if desired. E.g.,

```
 A.    NORTH AMERICA
+B.    SOUTH AMERICA
 C.    AFRICA
 D.    EUROPE
```

  When the student sees this list the + sign will not appear.

- (D) The Decision Frame

  There are three general forms for decision-type statements:

  (1) the"pattern" form, those that are based on a specific pattern of responses, (2) the "summary" form, those that are based on a summary of responses for selected frames and (3) the "computational" form, those that are based on relational comparisons of CALC expressions, functions and items. A decision statement always begins with IF and usually ends with

one or more commands F:, C: or B:. C:, if used, must be followed by a CALC statement. Every line must start with either IF, AND, or, ELSE,END or a command. Lines containing only commands will be subject to the IF statement immediately above. No parentheses are allowed to group AND or OR conditions.

The chart below shows optional construction of each of the three forms. Capitalized words indicate primitive words meaningful to PLANIT. Lower case words designate that appropriate substitutions must be made.

Pattern form:

```
      Optional Entries

                                                        AND
IF
AND   FROM frame, frame(s), tag(s)  frame(s), tag(s)  etc..OR
OR                                                  command
```

Summary form:

```
      Optional Entries

                   GR
IF                 GQ                 RIGHT                    AND
AND   FROM frame,  EQ    number       WRONG     frame(s)        OR
OR                 LQ                 SEEN                 command
                   LS                 MINUTES

                         ALL
                         NONE
       - - - - - - - - - - - - - - - - - - - -

IF
AND   FROM frame,  function name USED       frame(s)           AND
OR                                                              OR
                                                           command
```

Computation form:

```
                      GR
IF                    GQ                          AND
AND   expression      EQ      expression          OR
OR                    LQ                          command
                      LS
                      NQ
```

Notes:

Any inclusive block of frames can be written in the form, frame-frame (e.g. 4-7).

Any frame can be designated either by number or label.

The optional FROM primitive allows the LD to control the point in the records from which the search begins in the attempt to test the decision criteria. In the absence of the FROM, the search begins from the last entry of the current decision frame into the records or, if the current decision frame is being encountered for the first time, from the first entry made in the records.

ELSE and END are the two remaining primitive words that can start a line in a decision frame. The use of ELSE provides for the specification of an alternative set of actions that will occur should the preceding IF-statement turn out "false".

The END primitive terminates the conditions specified by the preceding IF-statement. It is usually optional since the next IF-statement automatically terminates the conditions of the previous one. However it is very useful if one desires to specify a set of unconditional actions to follow the execution of the immediately prior IF conditions within the same frame.

- (C) The Copy Frame
  The general form of the copy frame is C N where N is frame number or label. C N means to replicate the existing frame N at a new position in the lesson (determined by prior actions).

APPENDIX A

USING CALC STATEMENTS IN THE FRAME

```
FRAME 14.00 LABEL=*PLNTY

2.   TEXT.
*USE THE FUNCTION FTOC(X) TO CONVERT FROM DEGREES
*FAHRENHEIT TO DEGREES CENTIGRADE.  X CONTAINS DEGREES
*FAHRENHEIT AND FTOC(X) WILL PRODUCE DEGREES CENTIGRADE.
*GIVE ME 45 DEGREES FAHRENHEIT EXPRESSED IN CENTIGRADE.
*USE NUMBERS ONLY, DO NOT USE THE DEGREE SYMBOL.

3.   ANSWERS.
*0 FUNCTION FTOC(X)=(5/9)*(X-32)
*0 ALLOW FTOC
*0 CALC
*1+FTOC(45)
*

4.   ACTIONS.
*1 F:FINE, NOW WITHOUT USING THE FORMULA, COMPUTE 39F INTO CENTIGRADE.
*-R:

P/Q/M/D/C.
*Q

FRAME 15.00 LABEL=*

2.   TEXT.
*

3.   ANSWERS.
*0 PROHIBIT FTOC
*1+FTOC(39)
*

4.   ACTIONS.
*1 F:  B:AHEAD
*-R:
*
```

These last two frames working together show how the LD has a great deal of flexibility using CALC statements through the lesson and how he can control the use of functions. Remember that any answer prefixed by a number actually allows the lesson designer to follow it with any CALC statement. CALC statements prefixed by 0(zero) are done before the student answers. All other numbered (1-9) CALC statements are done in the order they appear (from top to bottom) after the student answers, and until a match is found. In frame 14, three CALC statements are done before the student answers: (1) the LD defines FTOC; (2) the LD allows the student use of it (indeed, if the student were to "←PRINT NAMES" (see CALC 10.1) FTOC would appear in his list); and (3) the LD puts the student in the CALC mode. This last step is a useful device. It allows the LD to turn on the CALC mode for the student. Therefore, the student can use CALC statements even if he forgets to strike the left arrow(←). The student must use ↑ or READY to enter his answer (i.e., to get back into the EX mode).

In the answer set, the LD uses FTOC for himself. Once the student gets it right, control falls through to the next frame (absence of B:) frame 15, where the LD does not even bother with a question having asked it in the feedback message of the previous frame (14). The first thing done is to PROHIBIT use of FTOC by the student to see if he can get the answer by himself. Naturally, the function FTOC is always available to the LD unless he DROP(S) it. Note also, at this time, had the student typed "PRINT NAMES" again, FTOC would no longer appear in his list.

Another interesting example of using CALC statements in the lesson follows.

```
P/Q/M/D/C.
*Q

FRAME 1.00 LABEL=*FACT

2.   TEXT.
*ENTER A NUMBER AND I WILL GIVE YOU THE FACTORIAL OF IT.
*

3.   ANSWERS.
*1 X=FACT(RESPONSE)
*

4.   ACTIONS.
*C:PRINT X R:ANOTHER NUMBER?
*#EX

ENTER A NUMBER AND I WILL GIVE YOU THE FACTORIAL OF IT.

*0
1.0

ANOTHER NUMBER?
*1
1.0
```

```
ANOTHER NUMBER?
*2
2.0

ANOTHER NUMBER?
*5
120.0

ANOTHER NUMBER?
*10
3628800.0

ANOTHER NUMBER?
*15
1307674368000.0

ANOTHER NUMBER?
*170
7.25741555*10**306

ANOTHER NUMBER?
*
```

This illustrates the use of two primitives of PLANIT, Viz: FACT and RESPONSE. FACT, if you will remember, was used before (frame 4). It is a built-in routine that will give you the factorial of any number from 0 to 170. RESPONSE has not been mentioned before. It is also a primitive of PLANIT and will always contain the student's last numerical answer. In the example given, the LD has combined them in a clever way to produce an interesting routine. After completing the frame, the LD entered "#EX" and this little lesson was executed. You see the results directly under "#EX".

APPENDIX B

AUTOMATIC TEXT REFORMATTING

The lesson designer (LD) has the option of letting PLANIT determine the number of words that compose each line of text. Using this option the LD can make insertions and deletions in his text very conveniently and PLANIT will adjust the format of the entire paragraph automatically. Two PLANIT features provide this option: (1) the use of the terminal dollar sign and (2) the fact that PLANIT will never allow print to pile up on the end of the line, but instead will separate the sentence at the end of the last word that will fit on the line and put the remainder of the sentence on the next line.

To use this option, simply end all lines of group 2 text (except the last one) with a dollar sign. Be sure to include a space between the last letter of one line and the first letter of the next, since they both may be printed on the same line.

Indicate new paragraphs by beginning the line with the backslash (\) carriage return indicator followed by the desired number of spaces for indentation.

APPENDIX C

TECHNIQUE USED FOR EVALUATING ALGEBRAIC EXPRESSIONS

With FORMULAS ON, PLANIT will attempt to match equivalent algebraic expressions. The technique is as follows:

1. PLANIT first searches the student's answer to see if all symbols (with the exception of numbers) in the LD's answer exist in the student's answer. If the student's answer has missing symbols, the answer is wrong. If the symbols are all present, PLANIT will go on to Step 2.

2. To all symbols in the LD's answer, it will assign successive prime numbers starting with 3, going left to right.

3. PLANIT will next perform the indicated arithmetic.

4. It will compare numerical answer with student's numerical answer (obtained by the same routine).

5. If the answers are the same numerically, student's answer is correct; otherwise he is wrong.

The following comparison will illustrate this process:

LD's ANSWER	POSSIBLE STUDENT ANSWERS	
A+(B+C)*D+5	a. A+(B*D+C*D)+5	right
	b. A+D*(C+B)+5	right
	c. 1+4+D*(B+C)+A	right
	d. D*(B+C)+5+3	wrong

If we go through the steps listed above, the LD's answer would work out as follows: First assign random prime numbers to A+(B+C)*D+5. Then perform the arithmetic making the same assignment to similar names which also appear in the student's answer.

Student Answers a, b, and c are fine because they have all the symbols and, when prime numbers are substituted for the same symbols as in the LD's answer, the numerical result is the same. Answer d is wrong because it has one symbol missing (even though the numerical result would come out the same).

APPENDIX D

PHONETIC ENCODING*

The technique used for encoding a word phonetically consists of four steps or passes, shown below. (The word upon which Steps B, C, and D operate is the output from the previous step.)

A. Letter equivalent. The first pass maps all letters into their letter equivalents (i.e., every letter in Row 1 is transformed into the letter immediately below it in Row 2). All other characters go through unchanged, e.g., ?→?, 5→5, #→#, etc.

1. ABCDEFGHIJKLMNOPQRSTUVWXYZ (original letter)
2. ABCDABCHACCLMMABCRCDABBCAC (letter equivalent)

B. The H replacement. With the exception of the first letter in any word, the second pass transforms each H in the word to the letter which precedes it.

C. Elimination of successive identical consonants. The third pass eliminates all but the first element of an uninterrupted sequence of a single consonant, e.g.:

SITTING becomes SITING
SUCCESSIVE becomes SUCESIVE
IRRESISTIBLE becomes IRESISTIBLE

D. Elimination of all A's. All vowels have been mapped into A (Step 1), the fourth pass eliminates these, and the final word has no vowels.

Examples:

ORIGINAL WORD:	IRRESISTIBLE	PHONETIC	FONETIC	BILLBOARD
Step A	ARRACACDABLA	BHAMADAC	BAMADAC	BALLBAARD
Step B	ARRACARDABLA	BBAMADAC	BAMADAC	BALLBAARD
Step C	ARACACDABLA	BAMADAC	BAMADAC	BALBAARD
Step D (final)	RCDDBL	BMDC	BMDC	BLBRD

The user must by now be aware of the danger in using the phonetic routine: Undesirable words can sneak through as phonetical equivalents. For example: THINK and THOMAS are phonetically equivalent; i.e., THINK → DMC and THOMAS → DMC. Therefore, if the LD is looking for THOMAS EDISON as the answer and the student types in I THINK IT WAS EDISON (KEYWORD ON in this case), he would get full credit. Nevertheless, the routine has shown itself to be extremely useful in many areas of practical concern and, when used judiciously, can be most helpful.

The maximum number of letters that the routine can handle in a given word is 16. Additional characters remain unchanged.

---

*Patterned in concept from article: Hewes, W. L., and Stow, K. H., Information Retrieval by Proper Name, Data Processing Magazine, June 1965, 18-22.

APPENDIX E

ADDITIONAL CAPABILITIES OF THE PROBLEM FRAME

I. Controlling the Sample Size Message

The standard message requesting the sample size (i.e., SAMPLE SIZE (MAX. XX)...) can be overridden by inserting a preferred message in Group 6 after the Y.

```
G6.  SS.
*  T  Y   HOW MANY SUBJECTS DO YOU WANT?
```

II. Controlling the Printing of the Data

Group 9 may be omitted from a problem frame. If omitted, the data will be set into VALUES and DATA but not put out on the teletype. Whenever Group 9 is encountered, the numbers in VALUES will be typed. Hence, the numbers in VALUES can be ranked or otherwise manipulated before they are given to the student via Group 9. The VALUES matrix can be set up through CALC and listed through Group 9. (See the RESET and RESTORE options.)

III. Data with Fractional Values

If rational numbers (not necessarily integers) are desired in the data, these can be specified by the use of a decimal point in the location characters in Group 9. The number of characters shown after the decimal point determines the fractional part to be retained, e.g.:

```
G9.  SH.
*    ITEM        BOYS       GIRLS
*       X        X.XX        X.XX
```

NOTE: Truncation of the random numbers in VALUES occurs in Group 9. If Group 9 is omitted from a problem frame, the numbers in VALUES will contain fractional parts.